EXHIBIT I

Redacted Exhibit 1 to the Declaration of Kevin Jeffay

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN JOSE DIVISION

CISCO SYSTEMS, INC., Plaintiff, v. ARISTA NETWORKS, INC., Defendant.	Case No. 5:14-cv-05344-BLF (PSG)

OPENING EXPERT REPORT OF KEVIN JEFFAY, PH.D. REGARDING INFRINGEMENT OF U.S. PATENT NO. 7,047,526

June 3, 2016

<u>HIGHLY CONFIDENTIAL – ATTORNEYS' EYES ONLY – SOURCE CODE</u>

TABLE OF CONTENTS

Page

I. INTRODUCTION 1

II. SUMMARY OF OPINIONS 1

III. BACKGROUND 2

A. Qualifications 2

B. Materials Reviewed 4

C. Compensation 4

D. Relevant Principles of Law 5

1. Claim Construction 5

2. Infringement (Direct, Indirect, DOE) 5

E. Level of Ordinary Skill in the Art 7

IV. TECHNOLOGY BACKGROUND 8

A. Switches and Routers 8

B. Background on Computer Networking 8

C. Command Line Interfaces 11

V. THE '526 PATENT 11

A. Summary of the '526 Patent and the Asserted Claims 11

B. The Detailed Description of the '526 patent 11

1. Multiple system administration and diagnostic tools could create complexities for system administrators. 12

2. The '526 patent's generic commands to operate OAM tools. 13

3. The Command Parse Tree in the Preferred Embodiment of the '526 Patent 15

C. The Asserted Claims of the '526 Patent 17

D. Claim Construction 20

E. Conception and Reduction to Practice 23

VI. ARISTA'S DIRECT INFRINGEMENT OF THE '526 PATENT 24

A. Operation of the Accused Products 25

1. Overview 25

2. Adding new commands 25

3. Capturing a user command 27

4. Command completion 27

5. Command Parsing 31

6. Invoking value functions 34

7. Sysdb and Agents 35

B. The Accused Products Include the Limitations of Claim 1 37

1. The Accused Products Meet the Limitations of the Preamble to Claim 1 37

2. The Accused Products Meet the Limitations of the Claim 1.1 41

3. The Accused Products Meet the Limitations of the Claim 1.2 45

4. The Accused Products Meet the Limitations of the Claim 1.3 52

5. The Accused Products Meet the Limitations of the Claim 1.4 56

6. The Accused Products Meet the Limitations of the Claim 1.5 58

C. The Accused Products Include the Limitations of Claim 2 62

1. The Accused Products Meet the Limitations of the Claim 2.0 62

2. The Accused Products Meet the Limitations of Claim 2.1 63

3. The Accused Products Meet the Limitations of the Claim 2.2 64

D. The Accused Products Include the Limitations of Claim 3 64

E. The Accused Products Include the Limitations of Claim 4 65

F. The Accused Products Include the Limitations of Claim 5 66

G. The Accused Products Include the Limitations of Claim 6 67

H. The Accused Products Include the Limitations of Claim 10 68

1. The Accused Products Meet the Limitations of the Preamble to Claim 10 68

2. The Accused Products Meet the Limitations of Claim 10.1 69

3. The Accused Products Meet the Limitations of Claim 10.2 69

4. The Accused Products Meet the Limitations of Claim 10.3 69

5. The Accused Products Meet the Limitations of Claim 10.4 69

6. The Accused Products Meet the Limitations of Claim 10.5 69

7. The Accused Products Meet the Limitations of Claim 10.6 70

I. The Accused Products Include the Limitations of Claim 11 70

1. The Accused Products Meet the Limitations of Claim 11.0 70

2. The Accused Products Meet the Limitations of Claim 11.1 70

J. The Accused Products Include the Limitations of Claim 12 70

K. The Accused Products Include the Limitations of Claim 13 71

L. The Accused Products Include the Limitations of Claim 14 71

1. The Accused Products Meet the Limitations of Claim 14.0 71

2. The Accused Products Meet the Limitations of Claim 14.1 71

3. The Accused Products Meet the Limitations of Claim 14.2 72

4. The Accused Products Meet the Limitations of Claim 14.3 72

5. The Accused Products Meet the Limitations of Claim 14.4 72

6. The Accused Products Meet the Limitations of Claim 14.5 72

M. The Accused Products Include the Limitations of Claim 15 73

1. The Accused Products Meet the Limitations of Claim 15.0 73

2. The Accused Products Meet the Limitations of Claim 15.1 73

3. The Accused Products Meet the Limitations of Claim 15.2 73

N. The Accused Products Include the Limitations of Claim 16 73

O. The Accused Products Include the Limitations of Claim 17 74

P. The Accused Products Include the Limitations of Claim 18 74

Q. The Accused Products Include the Limitations of Claim 19 74

1. The Accused Products Meet the Limitations of Claim 19.0 74

2. The Accused Products Meet the Limitations of Claim 19.1 74

R. The Accused Products Include the Limitations of Claim 23 75

1. The Accused Products Meet the Limitations of Claim 23.0 75

2. The Accused Products Meet the Limitations of Claim 23.1 75

3. The Accused Products Meet the Limitations of Claim 23.2 76

4. The Accused Products Meet the Limitations of Claim 23.3 76

5. The Accused Products Meet the Limitations of Claim 23.4 76

6. The Accused Products Meet the Limitations of Claim 23.5 76

7. The Accused Products Meet the Limitations of Claim 23.6 77

VII. INDIRECT INFRINGEMENT 78

A. Induced Infringement 78

1. Arista Customers Directly Infringe 79

2. Arista Had Knowledge of the '526 Patent 79

3. Arista Encourages and Instructs its Customers to Infringe 79

B. Contributory Infringement 80

1. Arista Knew that the Accused Products were Especially Made or Especially Adapted for use in the Infringement of the '526 Patent 81

2. The Accused Products have no Substantial Non-Infringing Uses 81

VIII. CISCO PRACTICES THE ASSERTED CLAIMS OF THE '526 PATENT 81

IX. NON-INFRINGING ALTERNATIVES 82

X. OTHER COMMENTS..83

I. INTRODUCTION

1. I have been asked by Cisco Systems, Inc., ("Cisco") to provide my expert opinions regarding the infringement and practice of claims of U.S. Patent No. 7,047,526 (the '526 patent)

2. I have been asked to analyze the products and/or services of Arista Networks, Inc., ("Arista"), including Arista's switches, routers, and associated products and services, that incorporate EOS, EOS+, vEOS network operating systems.

3. I have also been asked to analyze Cisco's products to determine whether they practice at least one asserted claim of the '526 patent.

4. My analysis, opinions, and reasoning are detailed below and in the attached exhibits, Exhibits 1-5, which I incorporate by reference into this report. If called to testify as to the contents of this report, I could and would testify competently thereto.

II. SUMMARY OF OPINIONS

5. It is my opinion that Arista infringes claims 1, 6, 10, 11, 13-16, 19, and 23 of the '526 patent in connection with the Accused Products.[1] Arista directly infringes each claim, either literally or through the doctrine of equivalents.

[1] The Accused Products include: Arista switches, routers, and associated products and services, that incorporate EOS, EOS+, vEOS network operating systems, including but not limited to Arista's 7010, 7048, 7050, 7050X, 7100, 7150, 7200, 7250X, 7280E, 7300, 7300X, 7500, 7500E Series products, including but not limited to 7504, 7508, 7316, 7308, 7304, 7280SE-72, 7280SE-68, 7280SE-64, 7150S-24, 7150S-52, 7150S-64, 7250QX-64, 7050QX-32, 7050QX-32S, 7050SX-64, 7050SX-72, 7050SX-96, 7050SX-128, 7050TX- 48, 7050TX-64, 7050TX-72, 7050TX-96, 7050TX-128, 7050S-52, 7050-64, 7050Q-16, 7050T-36, 7050T-52, 7050T-64, 7048T-A, 7010T, and 7500R. I reserve the right to opine on infringement of other Arista products that are variants of these products or include the same infringing functionality, as described in my report.

6. It is also my opinion that Arista infringes the claims 1, 6, 10, 11, 13-16, 19, and 23 indirectly, either by inducing others to infringe or by contributory infringement, as I understand those concepts (as described below).

7. It is also my opinion that certain of Cisco's products practice at least one claim of the '526 patent.

III. BACKGROUND

A. Qualifications

8. In forming my opinions, I am relying on my education and experience as described below.

9. I am a tenured professor in the Department of Computer Science at the University of North Carolina at Chapel Hill where I currently hold the position of Gillian T. Cell Distinguished Professor of Computer Science. I also currently serve as the Chairman of the Department.

10. I have a Ph.D. in computer science from the University of Washington, a M.Sc. degree in computer science from the University of Toronto, and a B.S. degree with Highest Distinction in mathematics from the University of Illinois at Urbana-Champaign.

11. I have been involved in the research and development of computing systems for over 30 years. I have been a faculty member at the University of North Carolina since 1989 where I perform research and I teach in the areas of computer networks, real-time systems, operating systems, multimedia networking, and network performance evaluation, among others. A major theme of my research has been aspects of router design and the measurement and analysis of traffic patterns and traffic performance in communications networks. My research

has also examined network support for multimedia applications. In my research I regularly use network switches and routers to construct and instrument experimental and production networks.

12. For example, in the late 1990s and 2000s my research evolved to consider router-based mechanisms for controlling the performance of network traffic. In much of this research my students and I built and instrumented network routers and performed large scale experiments with this equipment. Based on these experiments, in 2003, my group at UNC won the most prestigious research award for original research in computer networking. Much of my research has been performed jointly with industry.

13. I have authored or co-authored over 100 articles in peer-reviewed journals, conference proceedings, texts, and monographs in the aforementioned areas of computer science and others. I am currently an Associate Editor for the journal *Real-Time Systems* and have previously served as the Editor-in-Chief for the journal *Multimedia Systems*. In addition, I have edited and co-edited numerous published proceedings of technical conferences and have edited a book of readings in multimedia computing and networking (with Hong-Jiang Zhang) published by Morgan Kaufman. I am a co-author (with Long Le and F. Donelson Smith) of a monograph related to computer network protocols, and a co-author (with Jay Aikat and F. Donelson Smith) of a second monograph related to experimental computer networking.

14. I have served on numerous proposal review panels for the National Science Foundation and other international funding agencies in the aforementioned areas of computer science. I have served as a program chair or member of the technical program committee for over 100 professional, international, and technical conferences, workshops, and symposia.

15. I am a named inventor on three U.S. Patents. These patents are generally related to computer networking and the delivery of services over networks.

16. I have served as an expert witness and technical consultant in litigation matters concerning distributed systems, operating systems, multimedia networking, computer networks, cellular and wireline telephony, embedded systems and embedded software, real-time systems, among others. I have testified in several trials, arbitrations, and claim construction hearings as an expert witness.

17. I attach as Exhibit 1 my curriculum vitae, which includes a more detailed list of my qualifications.

B. Materials Reviewed

18. In forming my opinions, I am relying on my education and experience as described above.

19. Moreover, I have reviewed the '526 patent, including its file history and the references cited during prosecution, the Accused Products, source code for the Accused Products, documents produced by Arista related to the Accused Products. In addition, I have inspected Arista switches. I considered the operation of Arista switches in forming my infringement opinions described below.

20. I have also reviewed and considered the materials listed in Exhibit 2 of this Report.

C. Compensation

21. I was retained by Cisco's attorneys to be an expert witness for this lawsuit. Under my consulting agreement, I am paid my customary rate of $660 an hour for time spent on research, report preparation, deposition and/or trial. I am reimbursed for incurred expenses. I have not received, and do not expect to receive, any additional compensation for my work on this action, and payment of my fees is in no way contingent upon the outcome of this case.

D. Relevant Principles of Law

1. Claim Construction

22. I understand that claim construction is an issue of law for the Court to decide. I have been instructed by counsel that claim terms should be given their ordinary and customary meaning within the context of the patent in which the terms are used, i.e., the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention in light of what the patent teaches. Unless I expressly indicate otherwise, I will apply this ordinary and customary meaning to the terms of the claims throughout my analysis.

2. Infringement (Direct, Indirect, DOE)

23. I understand that an analysis of patent infringement requires two steps. The first step is to determine the proper meaning and scope of the asserted claims, as discussed above. The second step is to compare the claims, properly construed, to the accused devices or processes.

24. I understand that a person infringes a patent directly by making, using, selling, or offering for sale in the United States a product or method that embodies the patented invention. Such a person directly infringes the patent if he or she is the person who actually performs the act of making, using, selling, or offering for sale the embodiment of the patent in the United States.

25. I understand that to literally infringe a patent claim, i.e., to literally embody the claimed invention, a product or process must contain or embody each and every limitation of that claim, properly construed.

26. I understand that a product or process may be found to infringe the claims of a patent under the "doctrine of equivalents." Under the doctrine of equivalents, if a product or

process does not literally infringe, based upon the express terms of a patent claim, the product or process may nonetheless be found to infringe if the elements of the accused product or process are "equivalents" of the claimed elements of the patented invention. Such equivalency is typically found, I understand, if the elements of the accused product or process are "insubstantially different" from the claimed elements of the patented invention, and insubstantiality of difference typically occurs when the elements of the accused product or process perform the same function as the claimed elements of the patented invention, accomplish substantially the same result, and do so in substantially the same way.

27. I also understand that an analysis of the role played by each element in the context of the specific patent claim will help inform the inquiry as to whether (a) an accused substitute element matches the function, means, and result of the claimed element, or (b) the accused substitute element plays a role substantially different from the claimed element, because things that are equivalent in one context may be inequivalent in another context, and things inequivalent in one context may be equivalent in another context.

28. I also understand that there can be no infringement under the doctrine of equivalents if even one limitation of a claim (or its equivalent) is missing from the accused product or process.

29. I also understand that whoever actively induces infringement of a patent is liable as an infringer. I understand that in order to be liable for inducement there must be direct infringement of the asserted claim. I further understand that induced infringement requires that the infringer knew or should have known that his actions would induce actual infringements.

30. I understand that to be liable as a contributory infringer of a process or method claim, a party must import into the U.S. or offer to sell or sell in the U.S. a material or apparatus

for use in practicing a patented process and must know that the apparatus is especially made or adapted for use in infringement of the patent and is not a staple article suitable for substantial non-infringing uses.

E. Level of Ordinary Skill in the Art

31. I understand that a person of ordinary skill in the art is a hypothetical person skilled in the art, not a judge, a layperson, a person skilled in the remote arts, or a genius in the art at hand. This person of ordinary skill is an individual who is presumed to be one who thinks along the lines of conventional wisdom in the art and is not one who undertakes to innovate, whether by extraordinary insight or systematic research. This person of ordinary skill is also a person of ordinary creativity, not an automaton.

32. Relevant factors in determining the level of ordinary skill in the art include the educational level of the inventor and those who work in the field. Other considerations include various prior art approaches employed in the art, types of problems encountered in the art, the rapidity with which innovations are made, and the sophistication of the technology involved. Not all considerations may be present in every case, and the education level of inventors is not itself conclusive.

33. For the '526 patent, I believe that a person of ordinary skill in the art would have had a Bachelor's of science degree in electrical engineering, computer science or engineering, or a related field, and two to four years of work or research experience in the field of computer networking, or a Master's degree and one to two years of experience.

IV. TECHNOLOGY BACKGROUND

A. Switches and Routers

34. The '526 patent relates to command and interface control of executable software routines within software systems. The technology disclosed is particularly relevant to the interfaces to software operating systems controlling hardware devices such as switches and routers used to build computer networks.

B. Background on Computer Networking

35. In its most simple form, a computer network is a single cable that connects two or more computers together. Today, networks are typically formed by attaching computers to an interconnection device called a *bridge* or *switch*. A bridge is a device that allows other devices attached to it to exchange messages. Each bridge supports some number of attachment points for stations, called *ports*. When a message arrives at a bridge from a station (at an ingress bridge port), the message is "forwarded" (transmitted out an egress bridge port) to the destination station as described below.

36. When one computer wishes to communicate with another computer, hardware and software on that computer will create a message and transmit the message to the destination computer. The message will contain a set of addresses that specify its destination. As the message is transmitted to its destination, it may transit multiple bridges or switches. It is these devices that are responsible for getting the message to its destination. The addresses contained within the message will be used by the network devices to forward the message towards its destination until the message finally reaches the destination station.

37. Communication in a network proceeds according to a number of rules that control the format and processing of messages. Together, these rules define a *protocol* for

communication. In most networks, multiple protocols operating in concert are required to communicate messages between stations. These protocols are organized hierarchically as a series of hardware and software "layers" and are colloquially referred to as a "protocol stack." Layers in the stack are numbered and often referred to by their layer number. The layers are numbered from the lowest, most basic protocol layer, to the highest, most functional protocol layer. Historically, the two most dominant models of protocol layers are the OSI (Open Systems Interconnect) model and the "Internet" model. The OSI model defines a seven-layer protocol stack whereas the Internet model defines a simpler five-layer protocol stack. For our purposes here, the most relevant layers of the protocol stack are layers 1 through 3 (which are the same in both the OSI and Internet models).

38. The lowest layer, Layer 1, or "L1," is the *physical layer*. The physical layer protocol is typically implemented exclusively in hardware and is concerned with the low-level details of transmitting the binary data (the "1s" and "0s") that comprise the message over some physical medium. The physical layer refers to the physical media used to construct the network. Examples of different physical layers would be fiber optic cable, copper twisted pair wiring, or radio frequency spectrum. As these examples are each different physical media, each would require a different physical layer protocol because the process of transmitting binary data on the medium is different for each media type.

39. The next layer in the protocol stack, Layer 2, or "L2," is the *data link layer* (or simply the *link layer*). The link layer is responsible for transmitting a link layer message between two stations on the same network (*e.g.*, between two stations attached to the same bridge). The link layer uses the services of a Layer 1 protocol to transmit binary data using a message structure that is called a *frame.* A frame consists of a header and a payload. The payload is the

actual data being communicated across the network and may have additional, internal structure as described below. The header contains, among other things, a data link layer address of the station generating the packet (the "source address") and a link layer address of the destination station of the frame (the "destination address"). Transmission of a link layer frame may require changes to the header of the frame (as described in more detail below) but does not alter the substance of the communication.

40. Larger networks can be formed by interconnecting Layer-2 networks together. Two Layer-2 networks can be combined into a new, larger network by interconnecting the Layer-2 networks with a Layer-2 bridge or switch. However, for reasons that are beyond the scope of this tutorial, there are practical limits on the size of (i.e. number of stations on) a Layer-2 network. To get around these limitations, two Layer-2 networks may also be connected by a Layer 3 interconnection device to form an *internetwork* (often called a "routed network" or simply a "network"). Such Layer 3 interconnection devices are called *routers.* In the resulting network, an additional protocol, a Layer 3, or *network layer* protocol, is now required to communicate between stations on different Layer-2 networks.

41. The most common network-layer protocol in use today is the IP protocol (Internet Protocol) of the Internet. Layer 3 protocols solve the *internetworking* problem of interconnecting Layer-2 networks together. Routers are conceptually similar to bridges and switches except whereas bridges and switches use Layer 2 information to forward packets towards their destination, routers use Layer 3 protocol information for this purpose.

42. Today, network switching functions (Layer 2 forwarding) and routing functions (Layer 3 forwarding) are often present in network interconnection devices. These devices are commonly called "switch routers."

C. Command Line Interfaces

43. The command and interface control of the '526 patent are often referred to as "command line interfaces," or CLIs. The phrase is meant to connote a paradigm of interaction with a software system wherein the system is configured and/or controlled via entering or providing commands to the system that take the form of a line of text. The commands are typically generated by either a user (*e.g.*, by typing on a keyboard) or by a separate computer system that is in communication with the software system. The commands are provided to the software system one at a time via some interface, traditionally a serial or "terminal" interface. For each command entered, the software system typically would respond with one or more lines of text output.

44. Examples of command line interfaces to software systems include the "shell" found in operating systems, such as Linux, and the interfaces to many network switch and router operating systems.

V. THE '526 PATENT

A. Summary of the '526 Patent and the Asserted Claims

45. U.S. Patent No. 7,047,526, entitled "Generic command interface for multiple executable routines" was filed as application no. 09/604,880 on June 28, 2000, and issued on May 16, 2006. The inventors of the '526 patent are Jeffrey Wheeler and Paul Mustoe.

B. The Detailed Description of the '526 patent

46. The '526 patent generally relates to "command and interface control of Operating Administration and Monitoring (OAM) executable routines within software systems." '526 Patent at 1:7-9. The '526 patent describes that these "software systems" have "executable routines" such as, for example, network management tools. *See* '526 Patent at 1:10-27.

47. According to the '526 patent, "system administrators may attempt to utilize multiple tools within a software system in order to increase the available administration and diagnostic tools." '526 Patent at 1:28-30. Thus, "there is a need [to]…integrate [] command and control functionality for multiple programs, so the user does not need to learn the command formats and syntax for each program." *Id.* at 1:41-44. Accordingly, the invention of the '526 patent allows a user to utilize "a simple command language…for control[ing]…multiple [such] programs having respective command formats." *Id.* at 45-47.

48. Within this context, the '526 patent describes the use of a parser to interpret commands, such as those entered by a user. *See* '526 Patent at *Abstract*. A person of skill in the art would understand that these commands could be, for example, commands to configure or manage network equipment such as routers and/or switches.

49. The '526 patent allows a user to enter commands from a set of generic commands. When a command is entered, the parser interprets the command and determines (1) the correct management program to take action, and (2) the command for that management program that corresponds to the user-entered command. More specifically, the parser processes commands using a parse tree whose elements correspond to each element of a possible generic command, to determine a best match for the generic command entered by the user. The parser then issues a prescribed command for the management program.

1. **Multiple system administration and diagnostic tools could create complexities for system administrators.**

50. As computer systems and networks have become more complex and advanced, system administrators have employed multiple OAM tools to "increase the available administration and diagnostic tools for improved system performance." ('526 patent, 1:28–31.)

As the system administrator adds more and more such tools, some potential disadvantages become evident. Each OAM tool can have its own distinct functions and command syntax, and those distinct functions and syntax can be different than the commands that comprise the user-facing interface of the device. To successfully operate in this environment, a system administrator was required to have knowledge of the different function names and syntax formats of numerous commands for potentially numerous tools. ('526 patent, 1:31–37.) This increased the burden on system administrators to learn multiple different command formats and syntax for all the OAM tools in use. *Id.*

2. The '526 patent's generic commands to operate OAM tools.

51. The '526 patent solves this problem in the art by enabling a user to execute multiple management programs using a single interface. ('526 patent, 1:41–44.) It allows system administrators to control multiple administrative tools without requiring the system administrator to learn the syntax for each OAM tool, using one generic command interface. (*See* '526 patent, Abstract.) Specifically, the '526 patent provides a unified generic command interface that "incorporates the functionality of all external administrative executable binary files, RTM programs, agent manipulation scripts, and various requested snapshot queries, as well as including an extensive help system." ('526 patent, 3:21–27.)

The '526 patent provides system administrators with a single set of commands and syntax for multiple administrative and maintenance tools. Figure 1 (reproduced below) illustrates an embodiment of the '526 patent's system.

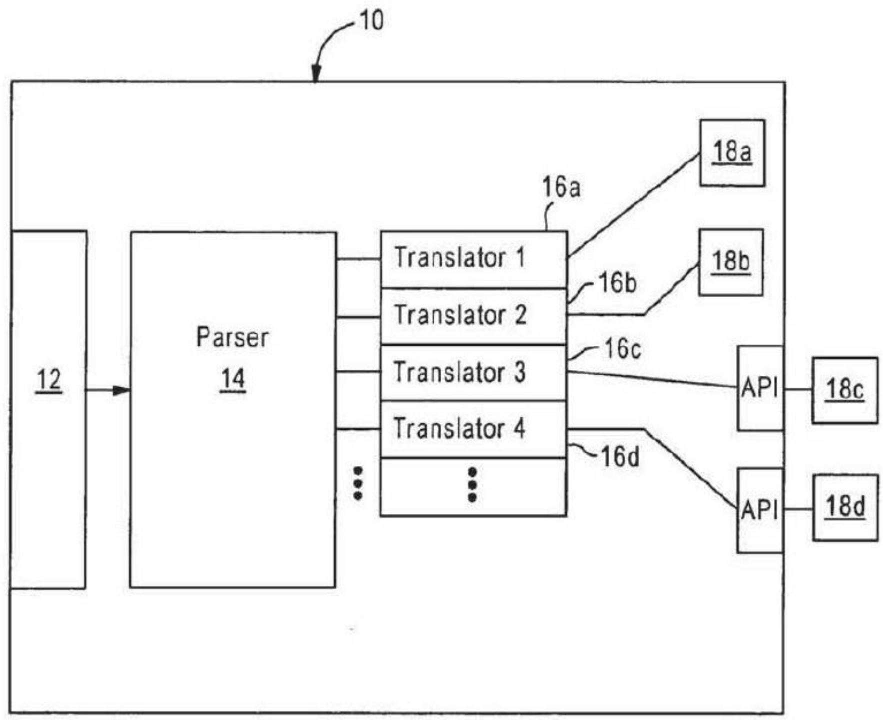


Figure 1

52. Figure 1 includes a user input interface 12, a parser 14, a plurality of translators 16 and a plurality of management tools 18 (*i.e.*, OAM tools). When a system administrator enters a generic command via interface 12, parser 14 validates the generic command. ('526 patent, 2:60-65.) Thereafter, translators 16 issue commands to respective management programs 18 according to respective command formats. ('526 patent, 2:60–3:1.) The parser 14 and the translators 16 "provide [system administrators with] a generic command syntax that integrates the functionality of the different [management] tools[,] and that automatically select[]the appropriate command for the best tool for executing a given generic command." (*See* '526 patent, 3:27–31.)

3. The Command Parse Tree in the Preferred Embodiment of the '526 Patent

53. To parse the generic commands provided by the '526 patent, the specification discloses the use of a "command parse tree." In the preferred embodiment, the command parse tree includes multiple elements and each element includes at least one corresponding generic command component and at least one corresponding command action value. ('526 patent, 1:51–54.)

54. A generic command may contain one or more input command words. The parser validates a received generic command by comparing each input command word to elements of the command parse tree. ('526 patent, 1:48–51.) The search of the command parse tree occurs by recursively traversing the tree and determining, for the generic command received, a tree element that best matches the generic command. ('526 patent, 3:51–61.) Once the system identifies the best matching tree element for the generic command, the selected management program issues a prescribed command based on the command action value of the identified tree element. ('526 patent, 1:54–58.)

55. Figure 2 shows an exemplary embodiment of the '526 patent's parser which includes the command parse tree 22 and a command word translation table 20.

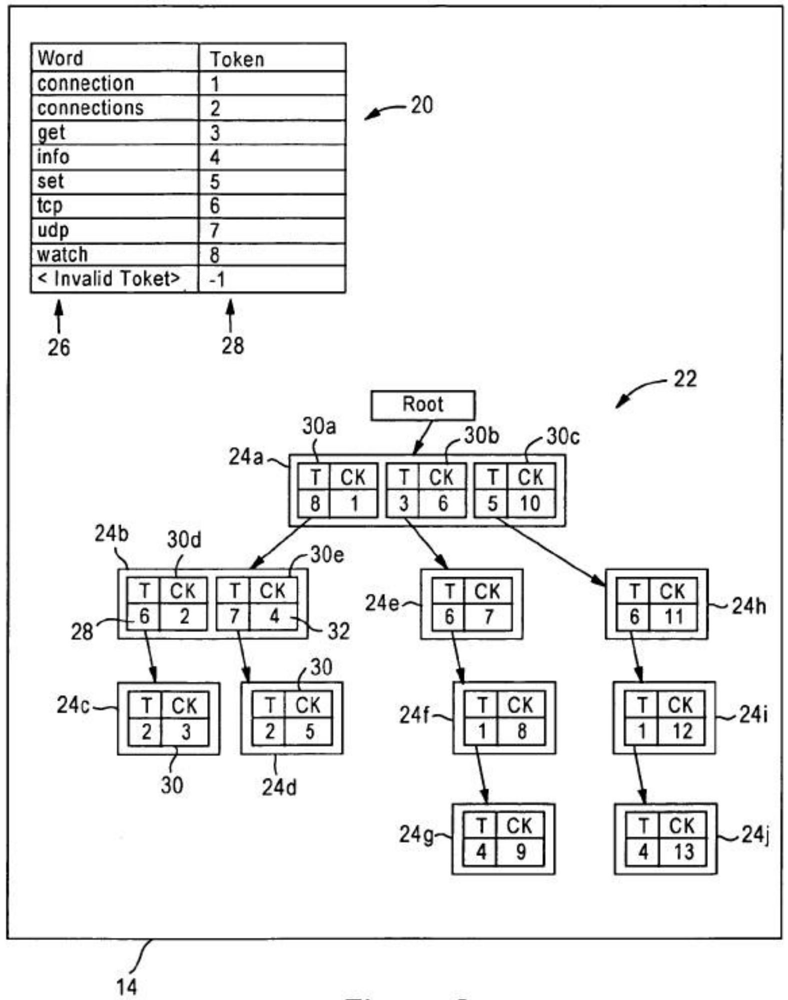


Figure 2

56. In this particular exemplary embodiment, the command word translation table 20 lists command words 26 that are valid according to the generic syntax. ('526 patent, 3:43–45.) Additionally, for each valid command word, the command word translation table also stores a corresponding token value 28. Parser 14 recursively traverses command parse tree 22 to identify tree elements that match a given generic command word. ('526 patent, 4:3–12.) The '526 patent's parser validates a received generic command by first identifying a command word and its corresponding token in the command word translation table. Parser 14 then compares the command word token to the command parse tree 22 to identify the best match. ('526 patent, 3:46–51.)

57. Also in this particular exemplary embodiment, each tree element 24 of the command parse tree includes at least one token-command key pair 30 that specifies a token T 28 and a corresponding command key CK 32. ('526 patent, 3:51–53.) The parser then uses the token-command key pair to identify the appropriate prescribed command for a management program. ('526 patent, 3:53–55.) If the parser determines that only a portion of the received generic command is valid (*e.g.*, only the first three command words are valid from a total of 4 command words received), the parser selects the command key 32 for the matching token 28 from the last valid tree element 24 (*e.g.*, the tree element matching the third valid command word). ('526 patent, 3:58–61.) Thus, even if the entire received generic command is not valid, a generic command still may be executed for a particular management program. (*See* '526 patent, 4:37–51.)

58. I understand the inventions of the '526 patent are not limited to this particular embodiment, consistent with my understanding of patent law and the disclosures found in the patent specification.

C. <u>The Asserted Claims of the '526 Patent</u>

59. The asserted claims are reproduced in their entirety below.

1. A method in a processor-based system configured for executing a plurality of management programs according to respective command formats, the method comprising:

receiving a generic command from the user;

validating the generic command based on a command parse tree that specifies valid generic commands relative to a prescribed generic command format, the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the validating step including identifying one of the elements as a best match relative to the generic command; and

issuing a prescribed command of a selected one of the management programs according to the corresponding command format, based on the identified one element.

2. The method of claim 1, wherein the generic command includes at least one input command word, the validating step including:[2]

comparing each input command word to a command word translation table, configured for storing for each prescribed command word a corresponding token, for identification of a matching token; and

determining a presence of the matching token within the command parse tree for each input command word.

3. The method of claim 2, wherein the determining step includes recursively traversing the command parse tree based on an order of the input command words for identification of the matching token within the identified one element.

4. The method of claim 3, wherein the issuing step includes issuing the prescribed command based on a corresponding command key specified for the matching token within the identified one element.

5. The method of claim 4, wherein the issuing step further includes accessing a prescribed translator configured for converting the generic command according to the corresponding command format into the prescribed command based on the corresponding command key.

6. The method of claim 5, wherein the validating step including validating at least a portion of the generic command by identifying the one element having the best match relative to the portion of the generic command, the issuing step including issuing the prescribed command based on the identified one element corresponding to the portion of the generic command.

10. A system configured for executing a plurality of management programs according to respective command formats, the system comprising:

a parser having a command parse tree configured for validating a generic command received from a user, the command parse tree configured for specifying valid generic commands relative to a prescribed generic command format and having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the parser identifying one of the elements as a best match relative to the generic command; and

---

[2] Italics indicates my understanding that the claims is are not currently independently asserted, but rather form part of another asserted claim.

a plurality of translators configured for issuing commands for the management programs according to respective command formats, the parser outputting a prescribed command to a selected one of the translators based on the identified one element.

11. The system of claim 10, wherein the parser further comprises a command word translation table configured for storing for each prescribed command word a corresponding token for identification of a matching token, the parser configured for determining a presence of the matching token within the command parse tree for each input command word.

12. The system of claim 11, wherein the parser recursively traverses the command parse tree based on an order of the input command words for identification of the matching token within the identified one element.

13. The system of claim 12, wherein the parser validates at least a portion of the generic command by identifying the one element having the best match relative to the portion of the generic command.

4. A computer readable medium having stored thereon sequences of instructions for executing a plurality of management programs according to respective command formats, the sequences of instructions including instructions for performing the steps of:

receiving a generic command from the user;

validating the generic command based on a command parse tree that specifies valid generic commands relative to a prescribed generic command format, the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the validating step including identifying one of the elements as a best match relative to the generic command; and

issuing a prescribed command of a selected one of the management programs according to the corresponding command format, based on the identified one element.

15. The medium of claim 14, wherein the generic command includes at least one input command word, the validating step including:

comparing each input command word to a command word translation table, configured for storing for each prescribed command word a corresponding token, for identification of a matching token; and

determining a presence of the matching token within the command parse tree for each input command word.

16. The medium of claim 15, wherein the determining step includes recursively traversing the command parse tree based on an order of the input command words for identification of the matching token within the identified one element.

17. The medium of claim 16, wherein the issuing step includes issuing the prescribed command based on a corresponding command key specified for the matching token within the identified one element.

18. The medium of claim 17, wherein the issuing step further includes accessing a prescribed translator configured for converting the generic command according to the corresponding command format into the prescribed command based on the corresponding command key.

19. The medium of claim 18, wherein the validating step including validating at least a portion of the generic command by identifying the one element having the best match relative to the portion of the generic command, the issuing step including issuing the prescribed command based on the identified one element corresponding to the portion of the generic command.

23. A system configured for executing a plurality of management programs according to respective command formats, the system comprising:

means for validating a generic command received from a user, the validating means configured for specifying valid generic commands relative to a prescribed generic command format and having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the validating means identifying one of the elements as a best match relative to the generic command; and

a plurality of translators configured for issuing commands for the management programs according to respective command formats, the validating means outputting a prescribed command to a selected one of the translators based on the identified one element.

D. <u>Claim Construction</u>

60. I understand the parties in this litigation held a claim construction hearing on April 8, 2016 regarding certain disputed terms for the '526 patent. Based on the Amended Joint Claim Construction Chart filed with the Court (Dkt. 216), I understand the parties' current proposed constructions are as set forth below:

Term	Cisco Proposal	Arista Proposal
"management programs"	"separate tools or external agents having their own respective command formats that provide management functions"	"tools that are configured to execute user-entered commands having their own respective command formats rather than the generic

		command format"
"generic command"	"command that provides an abstraction of the tool-specific command formats and syntax, enabling a user to issue the command based on the relative functions, as opposed to the specific syntax for a corresponding tool"	Indefinite, OR if not indefinite: "command having a format and syntax that is an abstraction of the command formats and syntaxes of more than one management program, as opposed to the specific syntax for any such management program"
"command parse tree"	: "a hierarchical data representation having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value"	"data structure consisting of linked nodes, with a root node (a node with no parent nodes), and where the remaining nodes are either a branch node (a node with a parent node and one or more children nodes), or a leaf node (a node with a parent node and no children nodes)" "command parse tree": "tree for interpreting commands where each node, or element, corresponds to one or more command components"
"the validating step including identifying one of the elements as a best match relative to the generic command"	Plain and ordinary meaning (except that specific terms appearing within the phrase should be construed as proposed above)	Indefinite, OR if not indefinite: "the validating step having the capability of both identifying the element in the parse tree that exactly matches the generic command, and, in the absence of an exact match, identifying the element that contains the last validated component of the generic command"
"the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value"	Plain and ordinary meaning (except that specific terms appearing within the phrase should be construed as proposed above)	"elements": "nodes" "command action value": "piece of data that uniquely represents the prescribed command." the entire phrase: "the command parse tree having

		nodes, such that each node specifies a unique command action value for each generic command component."
"means for validating a generic command received from a user, the validating means configured for specifying valid generic commands relative to a prescribed generic command format and having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value, the validating means identifying one of the elements as a best match relative to the generic command"	<u>Function</u>: validating a generic command received from a user <u>Structure</u>: Parser 14 in Figure 2, which includes the command word translation table 20 and the command parse tree 22, as described in 3:36- 61, and equivalents	<u>Functions</u>: (1) validating a generic command received from a user (2) specifying valid generic commands relative to a prescribed generic command format, (3) having elements each specifying at least one corresponding generic component and a corresponding at least one command action value, and (4) identifying one of the elements as a best match relative to the generic command. <u>Disclosed structure</u>: A processor executing a parser, and a corresponding memory storing a command parse tree, wherein the parser executes the algorithm of Figure 3, and wherein (1) each node of the command parse tree specifies one token and a corresponding command key; (2) the top-level nodes of the command parse tree represent all possible valid first words in the input command, second-level nodes represent all possible valid second words for each valid first word in the input command, and so on;

61. I also understand that the Court has yet to issue its claim construction order resolving the parties' claim construction disputes.

62. I have reviewed the parties proposed constructions, including the hearing transcript for the claim construction hearing where various compromise proposals were mentioned though not formally adopted by the Court.

63. Unless otherwise noted, my analysis for the asserted claims applies under both parties' proposed constructions. Where appropriate, I address the differences in the parties' proposed constructions for particular terms.

64. If or when the Court issues its claim construction order, I reserve the right to update my analysis to address those claim constructions.

E. Conception and Reduction to Practice

65. In my opinion, the '526 patent was conceived of and reduced to practice no later than November 17, 1999. My opinion is based on my analysis of Cisco documents, comparison with the '526 patent and review of '526 patent inventors' testimony.

66. I have reviewed "UM CLI For R4.1 / Internal Design" ("UMCLI ID") which is an internal Amteva Technologies / Cisco document describing the internal design of the UM CLI product which I further understand embodies the '526 patent. CLI-CSI-00023639-00023652 (attached as Exhibit 3). In my opinion, this document describes the '526 invention. Many parts of it are almost identical to the '526 patent itself. For example, on pages 5-7 (CLI-CSI-00023643-23645) is a table of commands that is virtually identical to Appendix Part A of the '526 patent. On pages 8-11 (CLI-CSI-00023646-23648) are a series of tables describing various commands that match almost exactly the series of tables in Appendix Part B of the '526 patent. The description of command parsing and command dispatch on p. 12 (CLI-CSI-00023650) is also consistent with the descriptions in the '526 patent specification.

67. Furthermore, both '526 patent inventors, Jeffrey Wheeler and Paul Mustoe, testified that the UMCLI ID document described the '526 invention. *See, e.g.,* Wheeler Depo. Tr. (11/12/15) at 151:13-152:8 and 67:11-21; Mustoe Depo. Tr. (5/24/16) at 26:21-29:11 and 66:23-70:1.

68. The metadata for UMCLI ID document indicates a last modified date of 11/17/1999 and a created date of 8/23/2000. It does not make sense for a document to be created after it was last modified, so the 8/23/2000 date is suspect. The first page of the document (CLI-CSI-00023639) provides a document history, where all of the dates provided are in 1999. In light of the descriptions provided there, it appears the UMCLI may have been implemented by as early as July 1999 (July 15, 1999 is listed as a "Post implementation update"). Nevertheless, based on the dates of the revision history, and the last modified date of 11/17/1999, it is my opinion that the '526 invention was conceived and reduced to practice no later than November 17, 1999.

69. Both inventors' testimony further corroborate this date and the timeline listed in the document history on the first page. *See, e.g.,* Wheeler Depo. Tr. (11/12/15) at 100:21-109:7; Mustoe Depo. Tr. (5/24/16) at 67:18-70:1 and 71:16-74:17.

VI. <u>ARISTA'S DIRECT INFRINGEMENT OF THE '526 PATENT</u>

70. It is my understanding that Cisco has accused a series of Arista's switches and routers, the Accused Products (defined above), of infringing the '526 patent.

71. In the following sections, I summarize: (1) the products Cisco accuses of infringing the '526 patent, and (2) aspects of the operation of the Accused Products.

A. Operation of the Accused Products[3]

1. Overview

72. This section broadly discusses "the life of a CLI command" along the following points:

- Adding the command and callback ("value function") to the parse tree.
- Capturing a user command.
- "Completing" an incomplete command
- Parsing the command.
- Invoking the value function.

2. Adding new commands

73. When a CliPlugin defines a new command, it invokes the Mode.addCommand() method, defined in [CliParser.py : 215-220]. The Mode class is defined in [CliParser.py : 128-509].

74. For example, the spanning tree protocol subsystem adds a command in […/src/stp/CLIPlugin/StpCli.py : 60]:

```
configMode.addCommand( (tokenStp, tokenForwardTime, forwardTime,
setForwardTime) )
```

75. There is one argument to this addCommand(): a tuple consisting of four items:

---

3 [redacted]

While I reference version 4.14.5.1 of the source code, this version of the source code is representative across all versions of EOS. *See* 5/13/16 Sweeney Depo., 452:3-5 ("Q. And so if I looked at the latest version of EOS source code, that would be representative of how all versions of EOS source code perform the functions that we've been talking about today in terms of how commands and tokens are parsed? A. Yes, I think the latest version of the code uses the same basic structures and algorithms that we've used since the beginning in our implementation of the CLI.")

- tokenStp […/src/stp/CLIPlugin/StpCliLib.py : 13] is a CliParser.KeywordRule
- tokenForwardTime […/src/stp/CLIPlugin/StpCli.py : 45] is a CliParser.KeywordRule
- forwardTime [.../src/stp/CLIPlugin/StpCli.py : 48] is a CliParser.RangeRule
- setForwardTime [.../src/stp/CLIPlugin/StpCli.py : 52] is a function (called a "value function") that is invoked when the entire command matches.

76. The addCommand() function is a member of the BasicCli.GlobalConfigMode class defined in [BasicCli.py : 1146-1167], which derives from ConfigModeBase defined in [BasicCli.py : 994-1122], which itself derives from CliParser.Mode defined in [CliParser.py : 128-509]. CliParser.Mode defines addCommand() in [CliParser.py : 215-220], which calls addCommandToRule on line [CliParser.py : 219].

77. CliParser.py defines addCommandToRule(parentRule, commandRule, …) (lines [CliParser.py : 82-118]), which creates a new Rule out of the input commandRule using the function rule(commandRule) defined in [CliRule.py : 1487-1509]. The return type of rule() depends on the type of the rule passed as a parameter. If a tuple is provided (as in the above example), then ConcatRule is returned; otherwise, if a list is provided, then an OrRule is returned. After invoking rule(), addCommandToRule() sets various flags for this Rule and adds the new rule (in this case a ConcatRule) to the modeRule (in this case an OrRule) by calling "parent |= r" (where 'r' is the return value of rule()) [CliParser.py : 118]. (Note that the "|=" operation is aliased to the OrRule.__ior__ function [CliRule.py : 956-958], which in turn calls OrRule.addRule() [CliRule.py : 959-979].)

3. Capturing a user command

78. There are multiple ways to receive a command in the Accused Products. As a concrete example, this section traces through a CLI session wherein a user provides input to the device. In the source code, the CLI's entry point is the main() function in [Cli.py].

(a) Creating a CLI session

79. The main() function in [Cli.py : 963-1140] allocates an instance of the MainCli class [Cli.py : 1067] and saves it to variable 'cli'. MainCli [Cli.py : 671-726] inherits from the Cli class. The Cli class [Cli.py : 83-575] creates a "session" and stores it, by calling self.session_ = BasicCli.session(initialModeClass, …) [Cli.py: 90] in its constructor. When MainCli calls Cli's constructor, it does so with initialModeClass set to BasicCli.UnprivMode.

(b) Reading user input

80. [redacted]

81. The second argument to readline -- self.complete -- is a completion callback, which will be invoked when the user inputs <TAB> or question mark. This transfers control over the function Cli.complete(self, key, cmd, pos) in [Cli.py : 175-185].

4. Command completion

82. The function Cli.complete() [Cli.py : 175-185] calls self._tryToCompleteCmd() if the last character is a <TAB> [Cli.py : 180], or, calls self._printHelp() if the last character is a question mark [Cli.py : 183].

83. Suppose, as an example, the user presses <TAB> (the processing for a question mark is similar). The self.complete() callback provided to CliInput.readline() will invoke _tryToCompleteCmd(self, cmd) [Cli.py : 273-282], which in turn makes the following two calls for command completion:

(a) _getCompletionLists

84. First, on line [Cli.py : 275] tryToCompleteCmd() calls self._getCompletionLists(cmd) [Cli.py : 233-244]. This function calls self.getCompletions_(), [Cli.py : 187-231], which takes the command line input until now (up to the <TAB>) and breaks up the command into full tokens and at most one partial token. (The last token is a partial token if the second-to-last character is not whitespace.)

85. On line [Cli.py : 207] Cli.getCompletions_ then calls session._getCompletions(tokens, partialToken, …). Note that the partial token is being treated specially, since it is passed separately from the rest of the tokens.

86. Recall from above that session_ is created in Cli's constructor, and is of type BasicCli.Session. Thus, self.session_.getCompletions() calls BasicCli.Session.getCompletions(), [BasicCli.py : 598-638], which in turn calls self.mode_.getCompletions(tokens, partialToken, …).

87. This mode_ variable corresponds to the current mode of the CLI session (that was is initialized to BasicCli.UnprivMode). (However, depending on how the CLI is invoked, the mode_ variable can be initialized to other values, such as BasicCli.EnableMode or BasicCli.GlobalConfigMode.) Each of these modes inherit from the Mode class defined in [CliParser.py : 128-509], and none of these modes define their own getCompletions() function. Thus self.mode_.getCompletions() results in a call to Mode.getCompletions().

88. Mode.getCompletions() [CliParser.py : 371-409] calls self._getCompletions [CliParser.py : 342-369]. This function creates a new parse context and parses ("inhales") all of the full tokens. Specifically, the function iterates through the tokens list and calls self._inhaleToken() on each token individually. All of the complete tokens are inhaled first so that the system knows how to interpret the partial token. The parse context variable is used to keep track of this process.

89. Once the full tokens are inhaled and the parse context is updated, the _getCompletions() function calls completions = self._completions(context, partialToken) [CliParser.py : 361]. At this point, the full tokens that were inhaled are "embedded" in the context. If there was no partial token and a full command was entered, then this function will add an end-of-line indicator as a completion to signify that the command entered was complete [CliParser.py : 362 - 363]. If there was a partial token, then self._completions()'s references to 'token' in the subsequent code refer to the partial token.

90. _completions(self, context, token) [CliParser.py : 444-447] calls self.instanceRule.completions(self, context, token) on line [CliParser.py : 447]. On line [CliParser.py : 146], Mode.instanceRule is set to an OrRule containing self.modeRule and universalRule. Each of GlobalConfigRule, UnprivMode, and EnableMode set self.modeRule to OrRule on lines [CliParser.py : 1148, 1332, 1274], respectively.

91. The OrRule [CliRule.py : 932-1186] derives from the Rule class, and defines completions() [CliRule.py : 1170-1186]. On line [CliRule.py : 1176], this function calls self.childCompletions() for each rule in the context. childCompletions() is defined starting on line [CliRule.py : 204] in the Rule class, which is defined in [CliRule.py : 932-1186].

92. childCompletions() returns child.completions(context.rulemap[child], mode, token). The variable "child" here is the individual rule being parsed from the context (*e.g.*, a "child" of an OrRule would be one of the rules being or'ed together). Thus, the childCompletions() function invokes the completions() function for an individual rule, and returns the completions.

93. For example, a ConcatRule is a common example of an OrRule's child. ConcatRule.completions(self, context, mode, token) is defined in [CliRule.py : 901-918]. This function calls completions.update(self.childCompletions (context, mode, token, r)) on line [CliRule.py : 912]. The same childCompletions code above will be invoked again, but in this case the "child" of the ConcatRule will be parsed.

94. For example, KeywordRule is a common example of a ConcatRule's child. KeywordRule derives from TokenRule. In TokenRule's constructor, the variable "matcher" is set (on line [CliRule.py : 571]) to KeywordMatcher, which is defined in [CliRule.py : 556-578]. In the TokenRule constructor [CliRule : 479-491], the variable TokenRule.matcher_ is set to this "matcher" [CliRule.py : 487]. This matcher is invoked in TokenRule.completions [CliRule.py : 543-554] on line [CliRule.py : 546]. CliRule.py also defines KeywordMatcher.completions() [CliRule.py : 602-609].

95. In summary, _tryToCompleteCmd() parses the line until the <TAB> and returns the completions, unguarded completions, and the partialToken, if any.

(b) <u>_getCommandSuffix</u>

96. Second, on line [Cli.py : 281], _tryToCompleteCmd() calls self._getCommandSuffix(unguardedCompletions, partialToken) [Cli.py : 246-271], which will complete the command (*i.e.*, expand it out) if there is only one match.

97. This completes the code path if the last character typed was a <TAB> to invoke the completion. If the last character was a question mark, then the logic is similar. On line [Cli.py : 183] the complete() function [Cli.py : 175] calls self._printHelp() [Cli.py : 356-391]), which calls self.getCompletions_, much like _tryToCompleteCmd().

98. The logic is identical from here on, and the parse tree is traversed in exactly the same manner as if <TAB> were pressed. After the completions are gathered, the "?" path diverges from the <TAB> path in that it can show help strings corresponding to the (partial) command, or print that the command is ambiguous or unavailable.

5. Command Parsing

99. After obtaining a command, either from a user or a separate device, each line is eventually passed to the CLI session's runCmd() function. This function tokenizes the input line and attempts to parse the command and execute the corresponding value function(s), searching through the mode hierarchy for a matching command, if necessary. In each mode in which the system attempts to run the command, it calls that mode's parse() command, which is inherited from the parent Mode class.

100. Mode.parse() [CliParser.py : 309-329] calls (at line [CliParser.py : 313]) parseAndExecute() [CliParser.py : 264-307]. parseAndExecute() iterates over each token and calls self._inhaleToken() [CliParser.py : 449-509] on each. _inhaleToken in turn calls self.instanceRule.inhale() on the tokens. The instanceRule is a variable set by each class that inherits from Mode, however, in each case, it is set to OrRule. Thus, for each token, OrRule.inhale() [CliRule.py : 1081-1161] is called and operates as described next.

(a) Creating "context" state

101. OrRule.inhale() calls self._beginAll() [CliRule.py : 1041-1079], which creates the "context" for this instance of the OrRule, and calls self.beginChild(context, r) for each "subrule" r of this OrRule.

102. Rule.beginChild() [CliRule.py : 188-196], makes a new context newCtx and stores this in its parent's context by invoking context.rulemap_[child] = newCtx. Next, Rule.beginChild() calls child.begin(), which is defined in each Rule subtype, and is responsible for initializing context state. For example:

- ConcatRule.begin() [CliRule.py : 822-825] sets context.state_ to the list of subrules that remain to be matched (the entire list of its subrules).
- OrRule.begin() [CliRule.py : 986-988] sets context.state_ to the set of subrules that could potentially be matched (all of the subrules)

103. Rules that do not define their own begin() function derive from Rule.begin() [CliRule.py : 197-199], which sets context.state_ to NoMatch.

(b) <u>Recursively try to inhale</u>

104. On line [CliRule.py : 1114], OrRule.inhale() next calls OrRule.tryToInhale() [CliRule.py : 1003-1039]. For each remaining subrule "child" in its context.state_, on line [CliRule.py : 1012] this function calls childInhale() with the child as input. childInhale() [CluRule.py : 200-201] calls child.inhale() which invokes the inhale() function of whatever type that child is.

105. For example, suppose the subrule is a ConcatRule. ConcatRule.inhale() [CliRule.py : 890-899] is similar in structure to OrRule.inhale() and on line [CliRule.py : 893] calls ConcatRule.tryToInhale() [CliRule.py : 853-888] which in turn calls (on line [CliRule.py : 867]) childInhale() which in turn calls child.inhale(). As before, this invokes the inhale() function

of whatever type that child is. For example, suppose that this subrule is a TokenRule. TokenRule.inhale [CliRule.py : 496-540] uses a matcher (used on line [CliRule.py : 514] and initialized within Rule's constructor [CliRule.py : 487]) to match the input "token" to the rule token.

(c) Update 'context' state

106. If the TokenRule successfully matches the input token to the rule token, then it updates its context state (from NoMatch to the matched value) and invokes its value function (if defined). The ConcatRule thereby removes this TokenRule from its context's list of subrules that remain to be matched. Finally, the OrRule keeps the ConcatRule in its context's list of subrules that could match.

107. If, the TokenRule does not successfully match, then it returns NoMatch, which ConcatRule will return in its inhale() function. The OrRule thus removes that ConcatRule from its context state's subrules. Specifically, on lines [CliRule.py : 1028-1033], tryToInhale() will add this ConcatRule to a list of "refusers." On line [CliRule.py : 1140] of inhale(), endChild() is called on each child in the list of refusers, and removes child from the context.rulemap_.

(d) Return from tryToInhale()

108. Finally, OrRule.tryToInhale() will return with NoMatch if there are no subrules that accept all of the tokens.

109. Picking up after OrRule.tryToInhale() (*i.e.*, in parseAndExecute [CliParser.py : 264-307], a method in the Mode class), after the function has inhaled all of the user-supplied tokens (lines [CliParser.py : 286-288]), it calls instanceRule.inhale(None) [CliParser.py : 292].

110. Like the tokens above, "None" is "percolated" through the OrRule (lines [CliRule.py : 1122-1128]) and the ConcatRule (lines [CliRule.py : 869-871]) and through the

TokenRule (lines [CliRule.py : 497-499]), but only through the list of subrules that remain in these rules' context states.

111. This "percolating" of "None" through the parse tree indicates that there are no more tokens left. ConcatRule successfully inhales a 'None' if its list of remaining subrules is empty (meaning that it has matched all of its tokens). If this happens, then it invokes its value function as described next.

6. **Invoking value functions**

112. Every class that inherits from Rule calls invokeValueFunction() after successfully "inhaling" a token.

- TokenRule calls invokeValueFunction() in inhale() on line [CliRule.py : 531]
- ConcatRule calls invokeValueFunction() in inhale() on line [CliRule.py : 898]
- OrRule calls invokeValueFunction() in inhale() on line [CliRule.py : 1134]
- MultiRangeRule calls invokeValueFunction() in inhale() on line [MultiRangeRule.py : 743]

113. Additionally:

- SetRule inherits from OptionalRule on line [SetRule.py : 90], and defines inhale() to explicitly call OptionalRule.inhale()
- QuotedStringRule [StringRule.py : 87] has an inhale() function that calls its value() function that calls invokeValueFunction().

114. Recall that, when adding commands with addCommand(), there is typically a single callback function ("value function") provided. For instance, recall the example from [.../src/stp/CLIPlugin/StpCli.py : 60]:

configMode.addCommand((tokenStp, tokenForwardTime, forwardTime, setForwardTime))

115. In this example, setForwardTime is the value function. Recall that the rule() function would take this tuple and create a ConcatRule out of it, and set that ConcatRule's value function to setForwardTime(). Thus, when the ConcatRule successfully inhales the final token ('eol'), then it will call setForwardTime().

116. Each of the other rules (tokenStp, tokenForwardTime, and forwardTime) also have a value function, but in this particular example, they would are set to 'None', and thus would not be invoked during parsing. However, the mechanisms are in place that it would be possible to set a value function for each of these tokens.

7. **Sysdb and Agents**

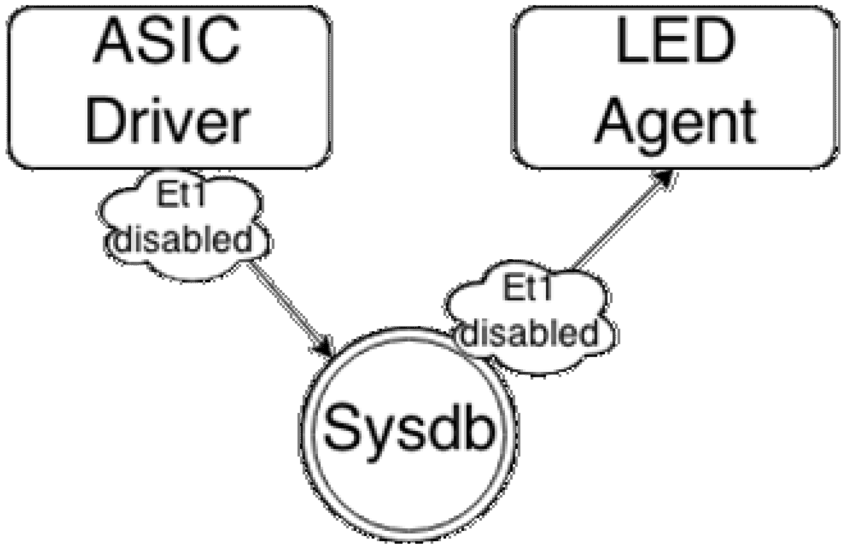
117. According to Arista documentation, Sysdb can be described as follows:

> Enter Sysdb, a small in-memory system that's somewhat like a database, or a file system, or a message bus, depending on how you see it. All the state that needs to be acted on by agents is stored in Sysdb, and each agent materializes a view of a subset of Sysdb that is relevant to its activities. For instance, the LED Driver cares about the status of the ports, but doesn't care about the routing table, therefore it loads only the portion of the state hierarchy that contains the interface statuses.

Arista GitHub: "Understanding EOS and Sysdb," available at https://github.com/aristanetworks/EosSdk/wiki/Understanding-EOS-and-Sysdb

118. The Arista documentation continues:

> The goal of Sysdb is to synchronize state across the different agents, so that the LED agent can receive a notification when ASIC driver marks an interface as "down". This process works in the following steps:



1. The ASIC driver notices that, say, Ethernet1's transceiver has been removed.

2. It then updates its local state hierarchy, setting interface/status/Ethernet1/oper_status tonot connected

3. The underlying agent infrastructure enqueues a state update message on its connection to Sysdb.

4. Sysdb reads the state update off of its end of the connection and updates its state hierarchy appropriately. At this point, both Sysdb and the ASIC driver know that Ethernet1 is not connected

5. Sysdb sends this state update to every other agent who has registered interest in that state update.

6. The LED agent, which registered interest in interface state at start-up, reads the update off of its connection to Sysdb and updates its copy of the interface state hierarchy.

7. This triggers a callback so the LED agent can switch off the little light, indicating that signal onEthernet1 was lost.

Arista GitHub: "Understanding EOS and Sysdb," available at https://github.com/aristanetworks/EosSdk/wiki/Understanding-EOS-and-Sysdb

119. When a CLI command creates a change in state by, for example, changing a configuration, state update notifications are propagated across the system, including to the agents

that are registered to receive updates on the particular table or change in state created by the CLI command.

120. Arista explains the following benefit for utilizing its Sysdb and agents architecture:

> Storing all the state in Sysdb also brings in another significant advantage: when an agent crashes or otherwise restarts, most of its state is preserved, by virtue of being in Sysdb. When the agent starts up again, it will reconnect to Sysdb, and retrieve its state however it was left in Sysdb before restarting. Additionally this restart is completely invisible to any other agent, since no two agents communicate directly. This means that agent crashes are significantly less severe, and that the software encoding the agent's logic can be seamlessly updated with no downtime.
>
> Arista GitHub: "Understanding EOS and Sysdb," available at https://github.com/aristanetworks/EosSdk/wiki/Understanding-EOS-and-Sysdb

B. The Accused Products Include the Limitations of Claim 1

121. I rely on my description of Accused Products described earlier in this report to further support my analysis of infringement of these asserted claims.

1. The Accused Products Meet the Limitations of the Preamble to Claim 1[4]

122. In my opinion, each of the Accused Products practices the following limitation [1.0] " method in a processor-based system configured for executing a plurality of management programs according to respective command formats, the method comprising . . ."

123. The hardware on which EOS resides is a processor-based system. For example, Arista 7500 Series Data Center Switch Data Sheet describes the 7500E Supervisor Module as utilizing processors, thus comprising a processor-based system as required by the preamble.

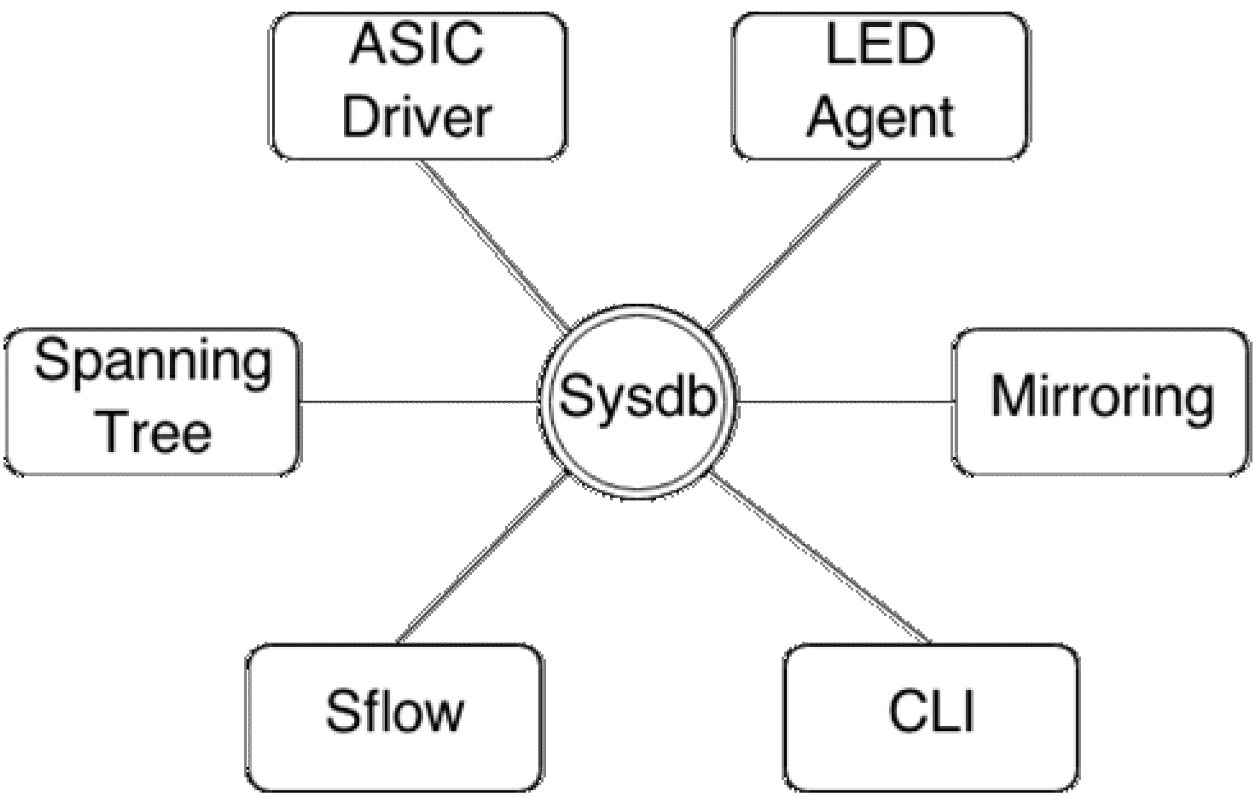
4 I am informed that the preamble of a claim, such as 1(a), may or may not be a limitation of a claim. For purposes of my analysis, I assume that the preambles of the asserted claims are limiting.

124. In addition, the Accused Products include processor-based system configured for executing a plurality of management programs. For example, EOS provides for processes or "agents" that allow interaction with external programs.

An introduction to EOS's architecture

EOS is composed of a number of processes, called "agents," each performing one specific role. These processes are just regular Linux processes. Even the "ASIC drivers", which manage the hardware chips that actually process packets, are userland processes.

Arista GitHub: "Understanding EOS and Sysdb"



Arista GitHub: "Understanding EOS and Sysdb," available at https://github.com/aristanetworks/EosSdk/wiki/Understanding-EOS-and-Sysdb

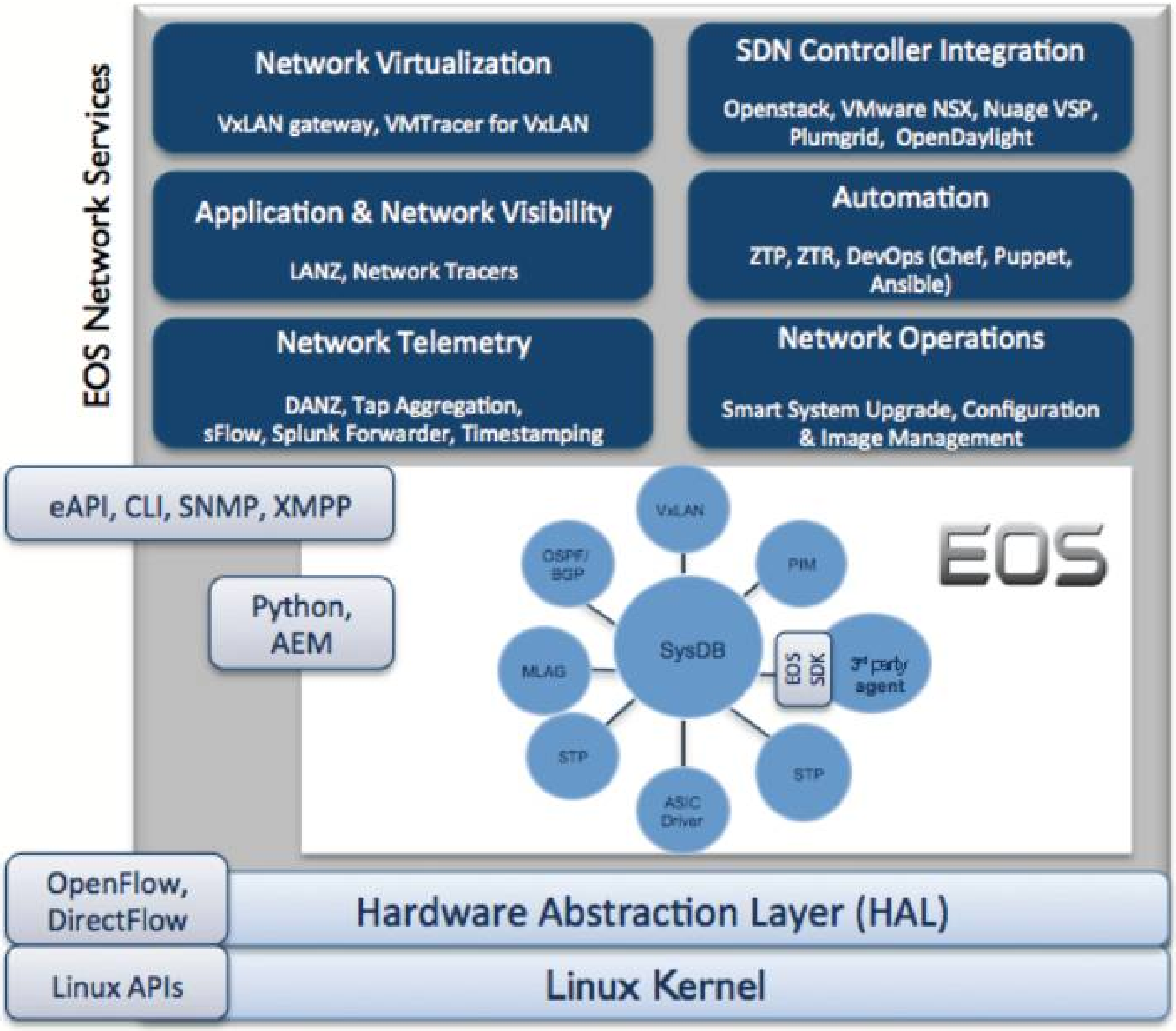


Figure 2: Arista EOS Architecture

Arista White Paper, "EOS: The Next Generation Extensible Operating System" (www.arista.com/media/system/pdf/EOSWhitepaper.pdf) at 3.

125. The agents and process described above, are in turn executed according to respective command formats. For example, the "show openflow" command results in the prescribed command OpenFlowShow --sysname=[system name] summary" issued by the agent in […/src/OpenFlow/CliPlugin/OpenFlowCli.py: 519-591]. Further examples are included in Exhibit 4.

126. [redacted]

127. [redacted]

[redacted]

128. As described by Mr. Sweeney, agents in EOS have their own virtual address space, are created and fail independently from other processes in EOS and "isolated" from other parts of EOS. Agents are thus separate tools or external agents from the CLI EOS parser and the Sysdb.

2. **The Accused Products Meet the Limitations of the Claim 1.1**

129. In my opinion, each of the Accused Products practices the following limitation [1.1] " receiving a generic command from the user."

130. For example, the command line interface receives generic commands using EOS's command line interface.

Command-Line Interface

The Extensible Operating System (EOS) provides the interface for entering commands that control the switch and manage the network. This chapter describes the command-line interfaces (CLI) that access the switch.

Arista User Manual 4.15.0F p. 85.

131. Examples of these generic commands include:

Generic Command
clear ipv6 neighbors [interface]
clear arp-cache [interface]
show tech-support
show platform pkt [detail]
show platform fm6000 tech-support
reload (and its variants; now, later, etc.)
show hsc status
show hsc detail ovsdb [database] [socket file name]
show kernel ip route [vrf <vrfName>]
show kernel ipv6 route
show kernel ip acl [vrf <vrfName>]
show kernel ipv6 acl
show kernel ip interface addr [vrf <vrfName>]
show kernel ipv6 interface addr
show kernel ip counters [vrf <vrfName>]
show kernel ipv6 counters [vrf <vrfName>]
show kernel ip arp [vrf <vrfName>]
show kernel ipv6 neighbors
show kernel interface counters [vrf <vrfName>]
show kernel interface addr [vrf <vrfName>]
logging follow
show openflow
show openflow ports
show openflow flows
show openflow statistics
show openflow queues
show openflow profiles

margin voltage (high | low | nominal) /* in diags mode */
session peer-supervisor [command]
show interface [<name>] counters queue [detail]
show interfaces [<name>] counters egress acl drop
show interfaces [<name>] counters queue bins
show interfaces [<name>] counters queue drops
show interfaces [<name>] counters queue ingress [detail]
show interfaces [<name>] counters replicated
show cpu counters queue
show platform petraA ip route [summary]
show platform arad ip route [summary]
show platform arad ip route vrf [default | vrfName]
show platform arad lem [diff]
show platform arad lem summary
show platform petraA arp
show platform arad arp
show platform petraA ipv6 route [summary]
show platform petraA ipv6 route tcam
show platform arad ipv6 route [summary]
show platform arad ipv6 route tcam
show platform arad ipv6 compression tcam
show platform arad vxlan vtep encapsulation
show platform arad mpls route
show platform arad ip pbr nexthop
show platform arad ip pbr nexthop-group
show platform arad vxlan mapping vsi [<vsi>]
show platform arad vxlan mapping vni [<vni>]
show platform arad vxlan vtep status [<vtep>]
show platform arad decap-group
show platform arad subinterface mapping [<subIntf>]
show platform schanaccel
show snmp mib
show snmp mib get
show snmp mib get-next
show snmp mib walk
show snmp mib table
show snmp translate <oid>

show platform trident l2mod-dma
show platform trident l2 hardware mac-address-table
show platform trident [<chip name>] mac-address-table
trace monitor <agent>
clock set <time> <date>
vm <name> console command
show uptime
show processes
show processes top
show processes top once
ip virtual-router mac-address <mac>
[default | no] [mls] qos map cos <0-7> to traffic-class <tc>
[default | no] [mls] qos map dscp <0-7> to traffic-class <tc>

132. As described above in Paragraphs 73-98, the CLI accepts these commands from the user, including the generic commands described above and then proceeds to parse the commands.

```
localhost(config)#
localhost(config)#show ?
  aaa                      Show AAA values
  agent                    Show agent settings
  aliases                  List all configured aliases
  arp                      ARP table
  banner                   Show system banners
  boot-config              Show boot configuration
  boot-extensions          Contents of boot extensions configuration
  clock                    Display the system clock
  dcbx                     Show IEEE DCBX information
  debugging                Show enabled debug messages
  diagnostic               Show diagnostic tests
  dot1q-tunnel             Show all enabled dot1q-tunnel ports
  environment              Show environment status
  errdisable               Show errdisable information
  error                    Show detailed information about an earlier error
  etherchannel             Synonym for show port-channel parameters
  event-handler            Show event-handler status
  event-monitor            Show event monitor logs
  extensions               EOS extensions present on this device
  file                     Show filesystem information
  flowcontrol              Show interface flowcontrol information
  history                  Display the session command history
  hosts                    Ip domain-name, nameservers and host table
  installed-extensions     Installed EOS extensions
```

Output from running "show ?" command

133. These generic commands are abstractions of the tool-specific command formats and syntax (described later as prescribed commands), which are selected by Cisco developers based on their subjective judgment and professional experience. The user is able to issue the generic command to access the function of the management program, rather than operating the management tool directly. Similarly, as shown by the difference in the commands in Exhibit 4, the generic commands are an abstraction of multiple management program command formats, rather than the specific syntax of any one management program.

134. These qualify as generic commands under either Cisco's or Arista's proposed claim constructions.

135. In Arista's proposed construction, the generic command is "an abstraction of the command formats and syntaxes of ***more than one*** management program." (emphasis added). In this case, the generic commands are abstractions of the multiple management programs LAG, STP, OSPF, and third-party agents described in Claim 1.5 below.

136. Arista's proposed construction also requires that there be some difference between the generic command and the syntax for the management program. Also as described in more detail in Claim 1.5 below, the syntax for generic commands in EOS is different than the syntax used in the accused management programs.

3. The Accused Products Meet the Limitations of the Claim 1.2

137. In my opinion, each of the Accused Products practices the following limitation [1.2] "validating the generic command based on a command parse tree that specifies valid generic commands relative to a prescribed generic command format."

138. As described above, the EOS CLI will parse the command entered by the user using a variety of rules, including OrRule rules and ConcatRule rules.

Concatenation of rules

```
CliParser.ConcatRule( subrules, ... )
```

Rule that matches the concatenation of a sequence of rules. ConcatRules are rarely constructed directly; they are usually constructed from rule expressions. The default value of a ConcatRule is the value of the rightmost subrule.

e.g.

```
CliParser.ConcatRule( tokenHost, tokenFilterHostname, name='host' )
```

Extending EOS CLI (https://eos.arista.com/extending-eos-cli/) p. 7.

Alternation of rules

```
CliParser.OrRule( subrules, ... )
```

Rule that matches precisely one of a set of rules. The default value of an OrRule is the value of whichever alternative matched.

e.g.

```
firstSecondOrThird = CliParser.OrRule( firstRule, secondRule, thirdRule )
```

Extending EOS CLI (https://eos.arista.com/extending-eos-cli/) p. 9

139. EOS validates input generic commands by parsing them with a plurality of different rules.

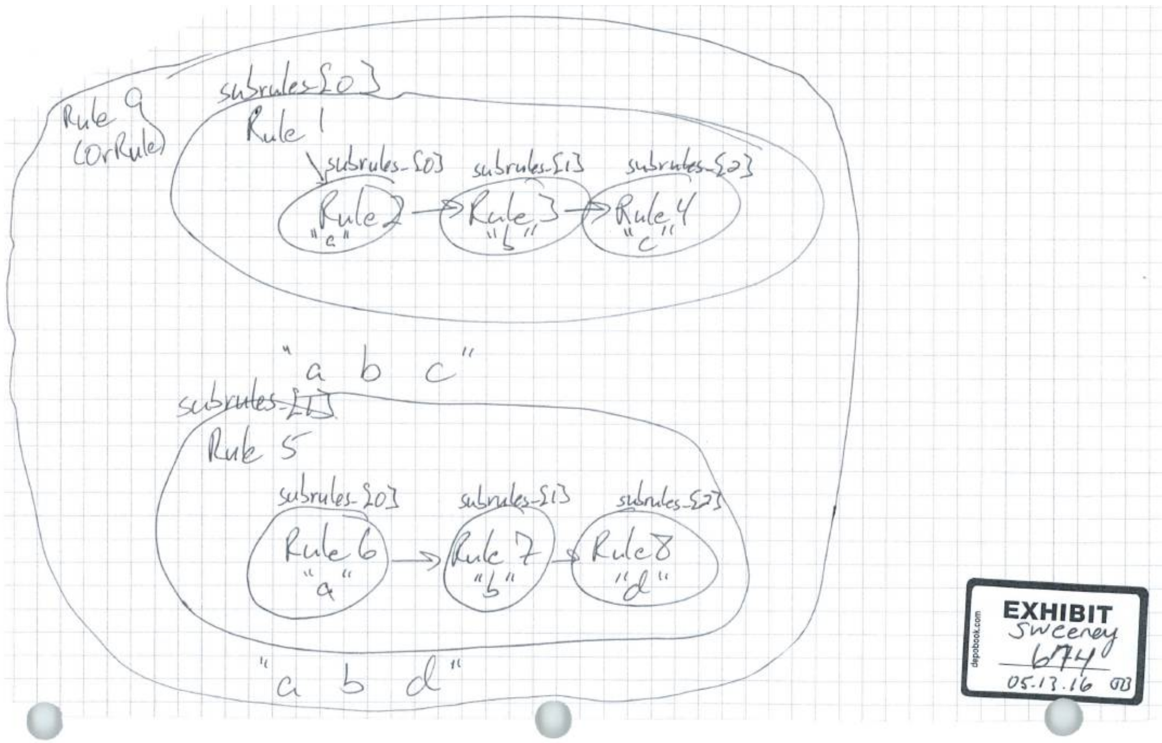
Rule definition

Rule types

- static match: TokenRule, with subclasses:
  - KeywordRule
  - PatternRule
  - RangeRule
  - FloatRangeRule
- dynamic match:
  - DynamicKeywordsMatcher
- concatenation of rules: ConcatRule
- alternation of rules: OrRule
  - DynamicNameRule
- wrapper rules: WrapperRule, with subclasses:
  - OptionalRule
  - HiddenRule
- iteration rules: IterationRule, with subclasses:
  - StringRule

Extending EOS CLI (https://eos.arista.com/extending-eos-cli/) p. 4.

140. When the CLI parser is in operation, the combination of those rules forms a command parse tree structure as required by the claim. Adam Sweeney, VP of Engineering at Arista and Arista's corporate witness on the operation of EOS, drew the following example depiction of the data structure created by the EOS CLI parser when parsing CLI commands:



141. It is well-known in computer science that one way to graphically depict a tree structure is as a set of nested circles as Mr. Sweeney did. For example, the same type of nested set structure is described as a tree in the seminal 1973 Knuth text:

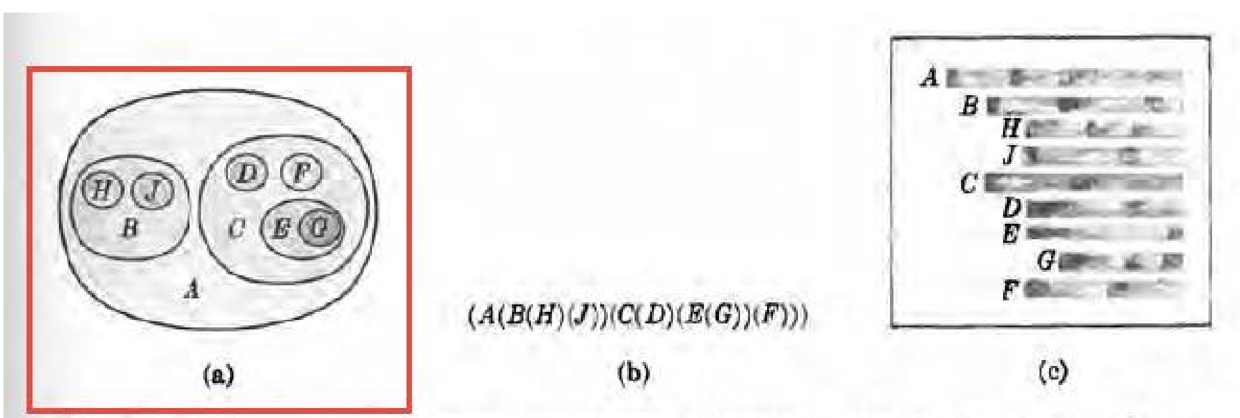
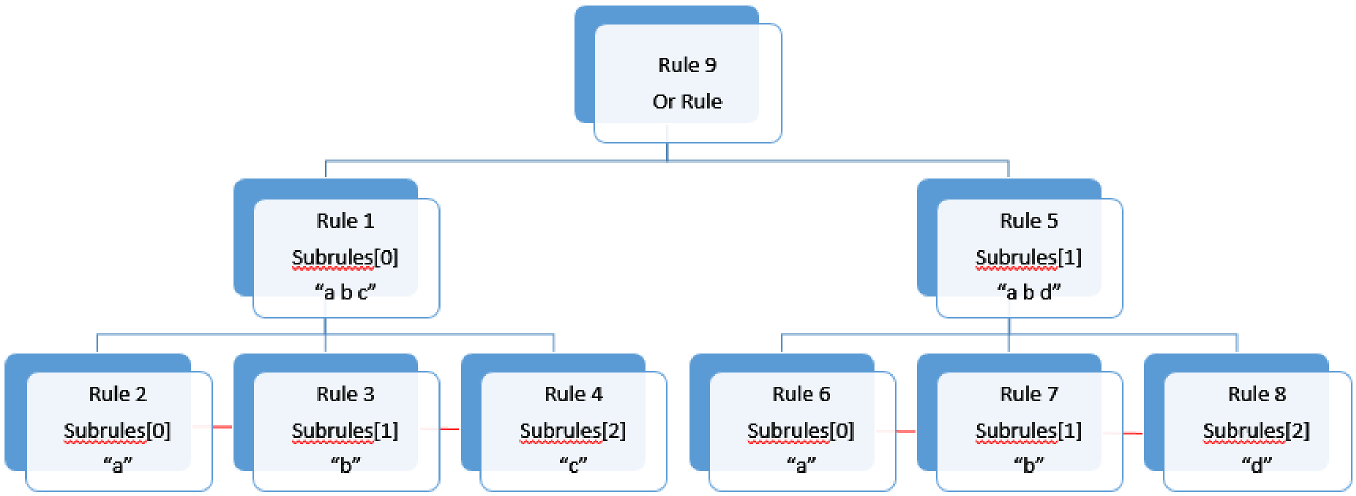


Fig. 20. Further ways to show tree structure: (a) Nested sets; (b) Nested parentheses; (c) Indentation.

Knuth, Donald. The Art of Computing Programming, Volume 1: Fundamental Algorithms (1973) (annotation added).

142. Consistent with this discussion, an alternative way to draw the same structure Mr. Sweeney created during his deposition would be the following:

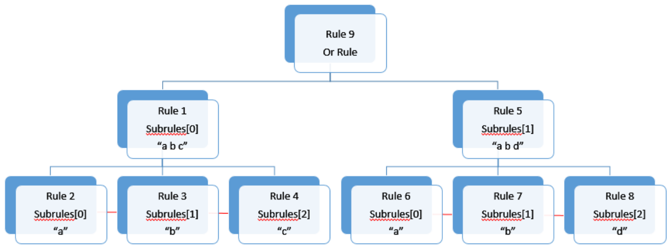
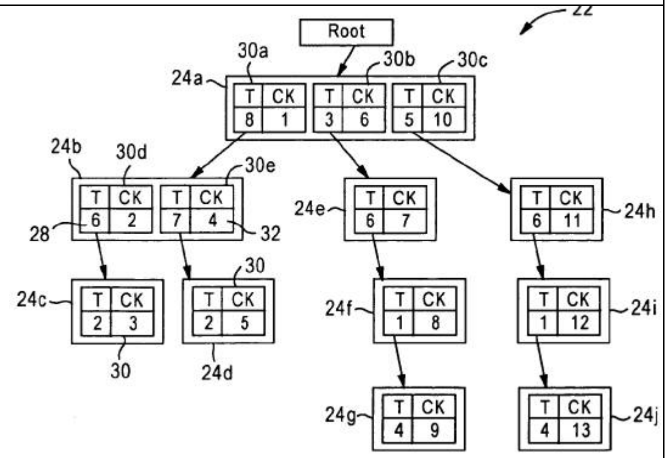


[5]

143. Thus, the command parse data structure created by the EOS CLI parser comprises a command parse tree. The nested rules, whether created using OrRule rules or ConcatRule rules, provide a hierarchical relationship that is traversed by the parser, as described above in Paragraphs 99-111. This parse tree is similar to that described in the preferred embodiment of the '526 patent.

---

[5] I've created a depiction of the command parse tree using the example commands Mr. Sweeney testified about at his deposition. This is an exemplary disclosure of the type of command parse tree structure within the EOS CLI. At trial, or when appropriate, I may provide additional command parse tree depictions consistent or complimentary to these depictions, including for the specific generic commands and prescribed commands discussed throughout this report.

<table>
<tr><th>EOS CLI Command Parse Tree</th><th>Command Parse Tree for Preferred Embodiment of '526 Patent</th></tr>
<tr><td>
</td><td>
</td></tr>
</table>

144. As described above, the parse tree is traversed in the following manner: Depending on the mode the user is in, the mode will call Mode.parse(). See CliParser.py :309. This in turn will call parseAndExecute : 264, which, for each token calls _inhaleToken(). Id at 449. EOS then calls instanceRule.inhale, which for the first rule is an OrRule.inhale CliRule.py :1081.

145. OrRule.inhale() will then populate the context state variable to include the list of subrules (as depicted above) which are still possible matches. The inhale() function then iteratively tries to compare the first token (*i.e.*, is the first word) to the subrules it has in the context state variable. That is, where the subrules start with a "show" token, then the subrules match at this stage. Where the subrules do not include "show" as the first token (*e.g.*, "clear"), they are eliminated from the context.state variable. At the end of this comparison, only those subrules that match the first token remain in the context.state variable.

146. As described above, the parser does this for each ConcatRule subrule available and for each successive token, until it: (1) finds only one match; (2) is still an ambiguous

command; or (3) does not find a match and issues an error. For the generic commands I've identified above, those are commands that properly parse and thus fall into category (1).

147. In traversing this command parse tree (*i.e.*, the rules, subrules and context.state described above), the EOS CLI parser validates the generic command based on a command parse tree that specifies valid generic commands relative to a prescribed generic command format, as the claim requires. The EOS CLI parser validates the generic command by parsing each individual CLI token to find a match until all tokens are parsed. It also specifies valid generic commands relative to a prescribed generic command format in that it parses the generic CLI commands described above according to their CLI format, and not according to any other format. The generic command format is the specific CLI syntax described for the generic commands described in Exhibit 4.

148. To the extent Arista argues that the structure of the rules in EOS CLI is not a "command parse tree" and does not literally infringe Claim 1.2, it is my opinion that EOS CLI infringes Claim 1.2 under the doctrine of equivalents. Even if the nested structure of the rules in EOS CLI were not a "command parse tree," it would infringe because it performs substantially the same function in substantially the same way to obtain substantially the same result. The function of the command parse tree is to arrange the components to be matched by the parser in a hierarchical way. One way to draw this hierarchy is depicted in paragraph 142, but a nested structure is another form to depict this hierarchy. Functionally, it shows the same hierarchical relationships between the components. The nested structure also allows parsing in substantially the same way. Because the nested structure captures the same hierarchical relationships between components, it allows traversal of the components according to this hierarchy. For example, whether the hierarchical structure is represented according to the diagram in paragraph 140 or in

paragraph 142, the parser could traverse all of the sub-trees of a given node before proceeding to another node. And the result obtained is the same regardless if the parsing elements are arranged in a tree or in a nested structure. In both cases, the parser is able to determine when which action to take depending on which nodes are traversed and matched

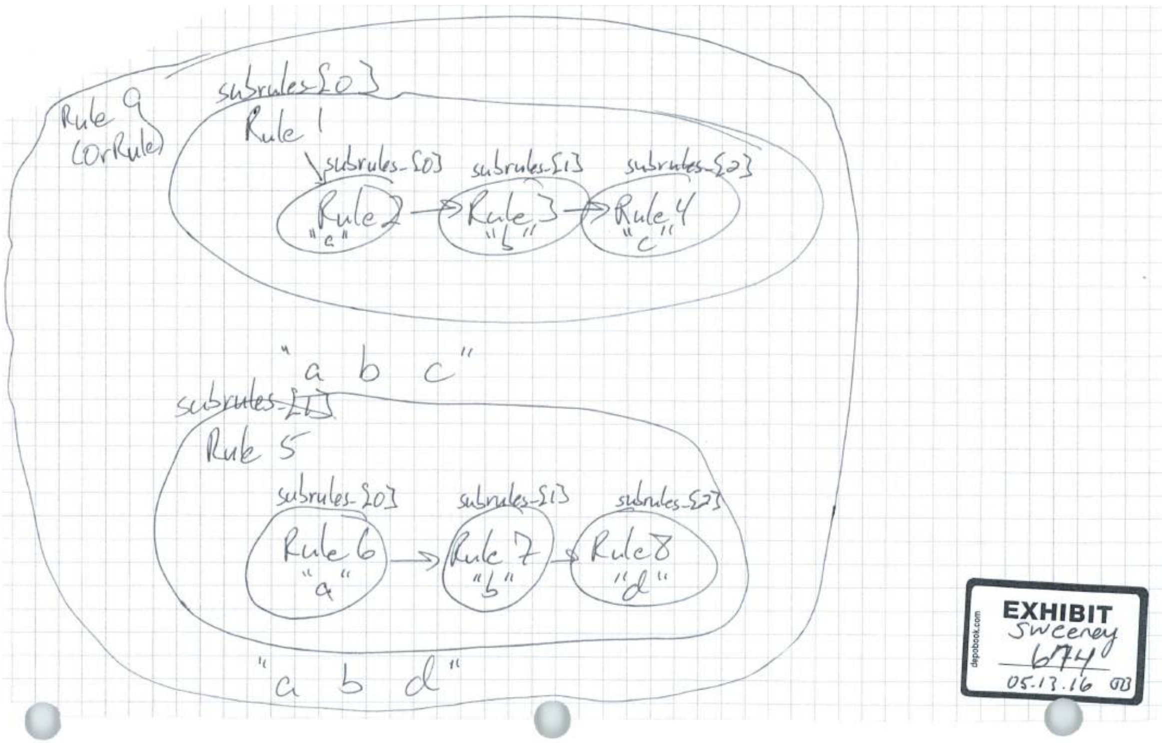
149. My infringement opinion remains the same under Cisco's or Arista's claim construction of "command parse tree." Arista proposes a construction where a tree is a data construction with root nodes, branch nodes and leaf nodes.[6] While I am not endorsing the notion that the "command parse tree" must have such a specific structure, the accused CLI EOS Command Parse Tree nonetheless has exactly this type of structure, as shown above in paragraphs 142 and 143. The OrRule rule is the root note in this tree, while the other remaining "Subrules" are either branch nodes (Rule 1 and Rule 5) or leaf nodes (other nodes), depending on whether they have children nodes or not, respectively.

4. The Accused Products Meet the Limitations of the Claim 1.3

150. In my opinion, each of the '526 Accused Products practices the following limitation [1.3] "the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value."

151. As described above, the EOS CLI parser in operate generates a command parse tree that includes the rules, subrules and context.state variable as described above. The parse tree, includes elements or nodes, as shown in Mr. Sweeney's drawing:

[6] I understand during the Markman hearing Arista agreed to withdraw its proposal for such a specific structure and agreed to "hierarchical data structure." Markman Hearing Tr. (4/8/16) at 74:6-76:6.



152. These elements, described as "subrules" in the source code, contain at least one corresponding generic command component in that they include a CLI token for the identified generic commands. Using the example drawn by Mr. Sweeney above, the subrule element/nodes in the tree include "a," "b" or other CLI tokens in each node. Thus, each element of the tree includes at least one corresponding generic command component.

153. Additionally, each element in the command parse tree includes a corresponding at least one command action value. As described above, when the CLI parser begins "inhaling" each CLI token, it builds a list of possible rules/subrules that could possibly match. As each CLI token is inhaled, the parser iteratively checks the state variable to eliminate those subrules that no longer match. When the parser finds a match and no other rules are left in the state variable, the parser then calls the value function for the CLI command rule in question.

154. Thus, the context.state variable, which contains the list of possible sub rules that still may be a match, is both contained in each element of the tree and contains command action

values—it contains a list of possible rules that are still potentially matches, each of which includes a value function. Thus, during each iterative step in traversing the command parse tree, context.state contains a reference to at least one command action value.

155. This is consistent with the preferred embodiment of the '526 patent, which for CLI tokens that are valid, but partial, allows matching to a command key so that the parser continues traversing the command parse tree:

> For example, assume that the parser 14 receives the valid command "watch tcp connections". The parser identifies the token value "8" as corresponding to the first command word "watch". The parser 14 than traverses the command parse tree 22 in step 42 to search for the matching token 28. As illustrated in FIG. 2, the parser 14 locates the matching token in the first tree element 24 *a*. If the parser 14 determines in step 44 that the first command word is valid, the parser 14 continues searching the next command word in step 46. If the first command word is invalid based on no match in the first element 24 *a* of the command parse tree, the parser 14 returns an invalid command message to the user in step 56.
>
> The parser 14 then parses the next word (e.g., "tcp") of the received generic command in step 46 by locating the corresponding token 28 (e.g., "6" for "tcp") in the table 20, and then traversing in step 48 the tree elements that depend from the matched tree element 24 *a* (e.g., 24 *b*). The parser 14 determines a match between the token 28 ("6") corresponding to the command word "tcp" in the token-command key pair 30 *d* in step 50, enabling the parser to continue for the next command word. As described above, the parser 14 repeats the process in step 52 for the third command word "connections" having the token "2" and identifying a match between the entire generic command and the token-command key 30 specified in the tree element 24 *c*. The parser 14 identifies in step 54 the prescribed command for a selected one of the translators 16 based on the value of the command key 32 within the matching token-command key pair 30 (e.g., "CK=3") of the last valid command word, which maps to a translation table that specifies a specific command for a specific translator 16.
>
> '526 Pat., 4:7-37.

156. As in the preferred embodiment of the '526 patent, the command parse tree within the EOS CLI parser includes, for each element, a variable which contains the still-possible matching CLI keyword rules, including their value functions.

157. In addition, each rule (which, in part, can comprise nodes/elements in the command parse tree) contains a "value" function that is invoked when the rule is being checked by the parser. For example, OrRule calls invokeValueFunction() in inhale() [CliRule.py : 1134] and MultiRangeRule calls invokeValueFunction() in inhale() [MultiRangeRule.py : 743]. Before an entirely valid command has been found, the value functions may be set to none, but all rules have value functions that are invoked. In this way, the value function is a value that can be used to identify a prescribed command and is contained in each element of the parse tree.

158. My infringement opinion also remains the same under Arista's proposed interpretation of the term "the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value." I understand that during the Markman hearing, Arista explained that they interpret the phrase to require that each generic command component must have a corresponding at least one command action value.[7] Even under this interpretation, the Accused Products still infringe. As explained in paragraph 143 to 149, 153 and 156-157, each "subrule" which can comprise nodes/elements in the command parse tree contains at least one corresponding generic command component. As explained in paragraphs 158 to 162 above, each "rule" which can comprise nodes/elements in the command parse tree, also contains a value. That is, each node/element in the tree has both a generic command component and a corresponding command action value.

[7] Markman Hearing Tr. (4/8/16) at 78:13-79:7.

5. The Accused Products Meet the Limitations of the Claim 1.4

159. In my opinion, each of the Accused Products practices the following limitation [1.4] "the validating step including identifying one of the elements as a best match relative to the generic command."

160. As described above, the EOS CLI parser will traverse the command parse tree. When the parser determines that it has found a valid command, it then executes the corresponding value function. *See* Paragraphs 99-116.

161. As another example, the EOS CLI parser includes autocomplete functionality to identify the element in the parse tree that is the best match for the received generic command.

- The command abbreviation **con** does not execute a command in Privileged EXEC mode because the names of two commands begin with these letters: **configure** and **connect**.

```
Switch#con
% Ambiguous command
```

- The command abbreviation **conf** executes **configure** in Privileged EXEC mode because no other command name begins with **conf**.

```
Switch#conf
Switch(config)#
```

Arista User Manual 4.15.0F p. 86.

See? Just like IOS, even down to the behavior of truncated commands being accepted, so long as they are not ambiguous. For example, while in configuration mode, entering just `ro` will not work because the CLI interpreter cannot figure out if I mean `route-map` or `router`:

```
Arista-1(config)#ro
% Ambiguous command
```

I can, however, find out what commands are available in one of two ways. First, I can hit question mark. This will give me a list of available commands that match what I've entered so far:

```
Arista-1(config)#ro?
route-map  router
```

I can also hit the Tab key, at which point the switch will respond with the longest match based on what I've typed so far:

```
Arista-1(config)#ro<TAB>
Arista-1(config)#route
```

Note that I did not type the word `route`; the switch inserted that via *autocompletion* when I hit Tab. At this point I decided that it's the `router` command I was looking for, so I added the `r`, and then hit question mark. The switch then recognized that `router` is a command, and listed the possible associated keywords:

```
Arista-1(config)#router ?
  bgp   Border Gateway Protocol
  ospf  Open Shortest Path First (OSPF)
```

I chose one of these protocols, after which the switch put me into protocol specific mode, and altered the command line to show where I was:

```
Arista-1(config)#router ospf 100
Arista-1(config-router-ospf)#
```

As with IOS, typing `exit` (or its nonambiguous abbreviation) got me out of the current level and popped me back up one level:

```
Arista-1(config-router-ospf)#ex
Arista-1(config)#
```

Arista Warrior at 60-61.

162. Thus, an exact string match for a valid command is not required, and EOS includes the ability to provide the "best match" using its autocomplete or attempted completion functionality. *See* Paragraphs 78-98.

163. My infringement opinion remains the same under either Cisco's or Arista's proposed construction of this limitation. Arista's proposed construction describes two situations: (1) the case of an exact match, and (2) the case in the absence of an exact match. The case referenced in paragraph 160 represents the case of an exact match. The case referenced in paragraph 162 represents the case in the absence of an exact match.

6. The Accused Products Meet the Limitations of the Claim 1.5

164. In my opinion, each of the Accused Products practices the following limitation [1.5] "issuing a prescribed command of a selected one of the management programs according to the corresponding command format, based on the identified one element."

165. As discussed above, the Accused Products, include the CLI parser, as well as management programs that are contained in different processes, such as agents. Management programs in EOS include LAG, STP, OSPF, and third-party agents. These qualify as management programs under either Cisco's or Arista's proposed constructions for "management programs."

Figure 30: Arista EOS Software Architecture showing some of the Agents

"Arista White Paper: Arista 7050X Switch Architecture ('A day in the life of a packet')" (http://people.ucsc.edu/~warner/Bufs/Arista_7050X_Switch_Architecture.pdf)

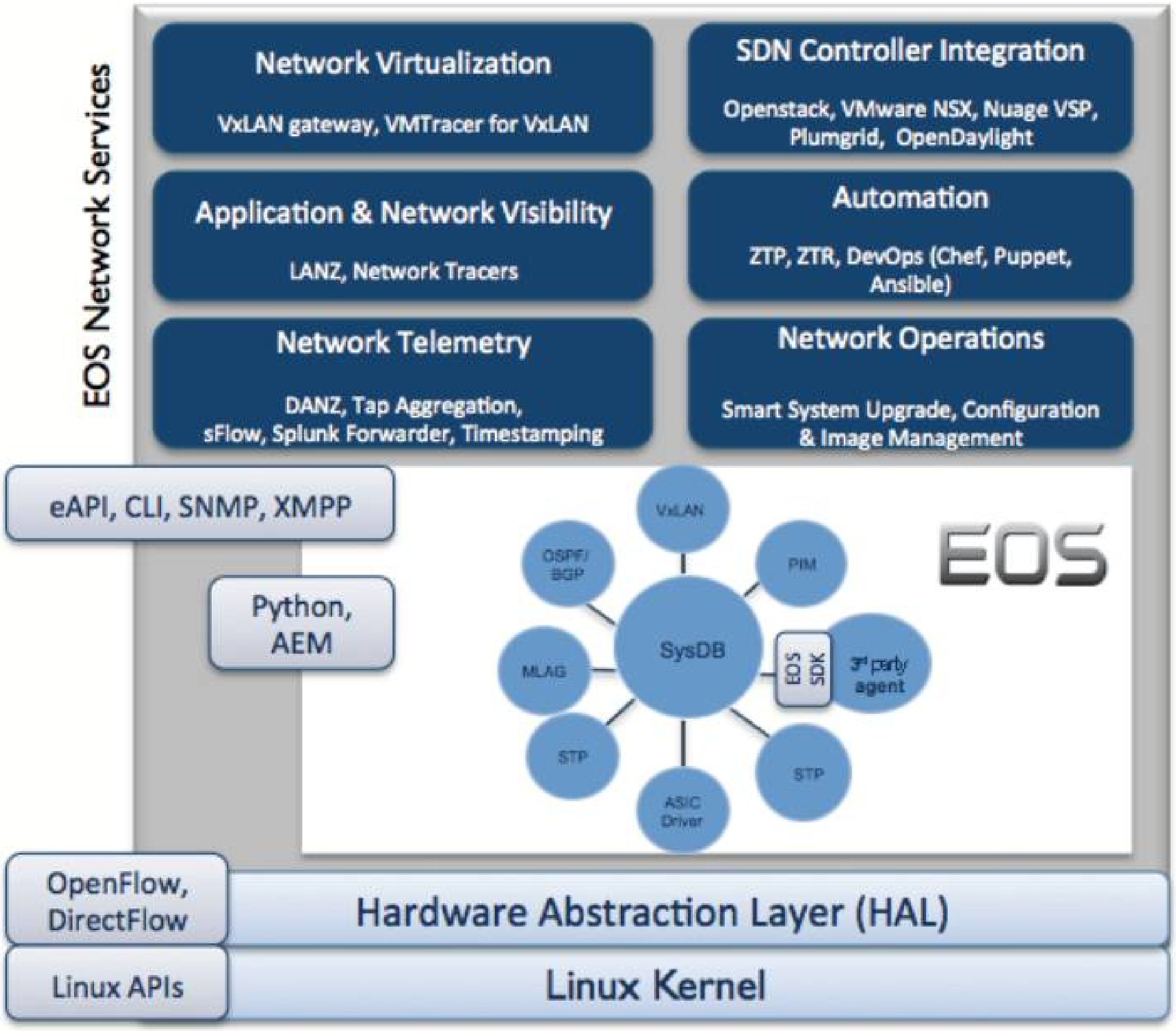


Figure 2: Arista EOS Architecture

Arista White Paper, "EOS: The Next Generation Extensible Operating System" (www.arista.com/media/system/pdf/EOSWhitepaper.pdf) at 3.

166. Arista's construction includes the idea of "execut[ing] user-entered commands," and hence infringe the '526 patent. All of these programs are able to accept user-entered commands. LAG is a Link-Aggregation Group and sample commands are shown on Arista's own EOS help page: https://eos.arista.com/how-to-configure-link-aggregation-groups-in-eos/ . STP is "Spanning Tree Protocol" and Arista's EOS has a specific "spanning-tree" command as

shown on: https://eos.arista.com/mlag-basic-configuration/. OSPF is accessible in Arista EOS through various user-entered commands as well. *See, e.g.,* https://eos.arista.com/tag/toi/.

167. All of these programs are also separate tools or external agents as required under Cisco's construction. As depicted in Arista's own diagrams, LAG, STP, OSPF and third party agents are shown as separate from the centralized Sysdb in Arista's EOS.

168. Arista's proposal also requires that there be some difference between the generic command the commands of the management program (Markman Hearing Tr. (4/8/16) at 57:11-23). In the case of the management programs identified above, they do not have the exact same syntax as the original EOS.

169. When the CLI executes a value function, the output will be sent to Sysdb. Sysdb then pushes that output to interested agents, which take action in response.

170. For example, SysdbMlag.py (ARISTA_SRC000224) and SysdEbra.py (ARISTA_SRC000169).

The event loop

The event loop is the core that your agent is built around: it is a constantly running loop that manages file descriptors, timers, and your agent's connection to Sysdb. When your agent writes a piece of state, the SDK immediately transforms that state to an internal representation and writes that state to a local copy of the state hierarchy. This write is added to an event queue. After you return control to the event loop, the write queue is drained and events are serialized to Sysdb. Sysdb then pushes the new state to all other interested agents, which can asynchronously react to the state to reprogram hardware, recalculate topologies, or simply set some other state in response.

Similarly, when Sysdb pushes out a notification about a state update to *your* agent, the event loop is notified and reads the state update off of the message queue. The SDK then transforms the message data into a stable representation and calls any of your agent's handlers that have subscribed to that state.

"Lifecycle of an SDK Agent" (https://github.com/aristanetworks/EosSdk/wiki/Lifecycle-of-an-SDK-Agent)

171. The generic commands identified above, through Sysdb, cause a prescribed command of a management program in a management format for that program to be issued, based on the matched element of the parse tree. For example, when the generic command "show openflow" is matched by the EOS Parser, in the OpenFlow plugin the "OpenFlowShow --sysname=[system name] summary" prescribed command is issued in …./src/OpenFlow/CliPlugin/OpenFlowCli.py. In contrast to the "show openflow" generic command, the "OpenFlowShow . . ." command is in the management format for the OpenFlowShow management program.

172. To the extent Arista argues that outputs sent to Sysdb are do not literally meet Claim 1.5, they infringe under the doctrine of equivalents. An output sent to Sysdb performs substantially the same function in substantially the same way and obtains substantially the same result as issuing a prescribed command. The function of issuing a prescribed command is to cause an action to happen in the management program. An output to Sysdb will also cause the desired action to happen in the management program because the agents, or management programs, respond based on the values of certain quantities in Sysdb. An output to Sysdb works in substantially the same way as issuing a prescribed command: in both cases, an action originating from the parser (the output to Sysdb or the issuance of a prescribed command) causes an action to occur in the management program. The result from an output to Sysdb or issuing a prescribed command is substantially the same. In the case of output to Sysdb, agents detect the change and perform the desired action. In the case of a prescribed command, the management program recognizes the prescribed command and performs the desired action.

C. The Accused Products Include the Limitations of Claim 2

1. The Accused Products Meet the Limitations of the Claim 2.0

173. In my opinion, each of the Accused Products practices the following limitation [2.0] "The method of claim 1, wherein the generic command includes at least one input command word . . ."

174. As described above, when receiving a generic command, the Accused Products, running EOS, receive generic commands. Those generic commands include at least one input command word, referred to as a token in the CLI code. *See*, e.g., CliRule.py, ln.4-26. (ARISTA_SRC000074).

show aaa

show boot-config

show class-map type qos] [<cmapName>]

show cvx

show mlag [detail]

show arp vrf <vrfName>] [resolve] [...] [summary]

show interfaces

EOS Command API Guide v.4.13.6F at 1-53.

```
localhost(config)#
localhost(config)#show ?
  aaa                     Show AAA values
  agent                   Show agent settings
  aliases                 List all configured aliases
  arp                     ARP table
  banner                  Show system banners
  boot-config             Show boot configuration
  boot-extensions         Contents of boot extensions configuration
  clock                   Display the system clock
  dcbx                    Show IEEE DCBX information
  debugging               Show enabled debug messages
  diagnostic              Show diagnostic tests
  dot1q-tunnel            Show all enabled dot1q-tunnel ports
  environment             Show environment status
  errdisable              Show errdisable information
  error                   Show detailed information about an earlier error
  etherchannel            Synonym for show port-channel parameters
  event-handler           Show event-handler status
  event-monitor           Show event monitor logs
  extensions              EOS extensions present on this device
  file                    Show filesystem information
  flowcontrol             Show interface flowcontrol information
  history                 Display the session command history
  hosts                   Ip domain-name, nameservers and host table
  installed-extensions    Installed EOS extensions
```

Output from running "show ?" command.

2. The Accused Products Meet the Limitations of Claim 2.1

175. In my opinion, each of the Accused Products practices the following limitation [2.1] "the validating step including: comparing each input command word to a command word translation table, configured for storing for each prescribed command word a corresponding token, for identification of a matching token . . ."

176. As described above, users enter commands using a CLI, which are in turn parsed by the EOS CLI parser. The parser, using the CliRule, BasicCli and CliParser classes translates the keyword commands into individual CLI tokens. Those tokens, and the translation process included within the EOS CLI parser, compare the input command word (e.g., "show") to a particular stored token within one or more rules, which the parser then identifies. When in operation, this parsing process embodies a table that translates commands in that it contains the

corresponding tokens and is paired with prescribed commands and management programs as described for claim 1. *See* Paragraphs 99-116. In other words, a command translation table.

177. To the extent Arista argues that the EOS CLI parser does not contain a command translation table, and therefore does not literally infringe Claim 2.1, it is my opinion that EOS CLI parser infringes under the doctrine of equivalents. EOS CLI performs substantially the same function as the limitations of Claim 2.1 in substantially the same way to obtain substantially the same result. The function performed by EOS CLI is substantially the same as that performed by a command translation table. In both cases, the function is to match command words to tokens. The way EOS CLI performs this function is substantially the same as a command translation table. In EOS CLI, the matching is achieved through the rules in various classes as described in paragraph 60 above. If a translation table is used, EOS CLI still uses the rules to perform the matching. The result of the EOS CLI process is the same as that for a command translation table. In both cases, if there is a matching token for the command word, it is found. .

3. The Accused Products Meet the Limitations of the Claim 2.2

178. In my opinion, each of the Accused Products practices the following limitation [2.2] "determining a presence of the matching token within the command parse tree for each input command word."

179. As described above, the EOS CLI parser in the Accused Products determine the presence of a matching CLI token while traversing the command parse tree for each command word. *See* Paragraphs 99-116, which are hereby incorporated by reference.

D. <u>The Accused Products Include the Limitations of Claim 3</u>

180. In my opinion, each of the Accused Products practices the following limitation [3.0] "The method of claim 2, wherein the determining step includes recursively traversing the

command parse tree based on an order of the input command words for identification of the matching token within the identified one element."

181. When traversing the command parse tree as described for claim 1, the EOS CLI parser does so recursively, based on an order of the input command words for identification of the matching token within the identified one element.

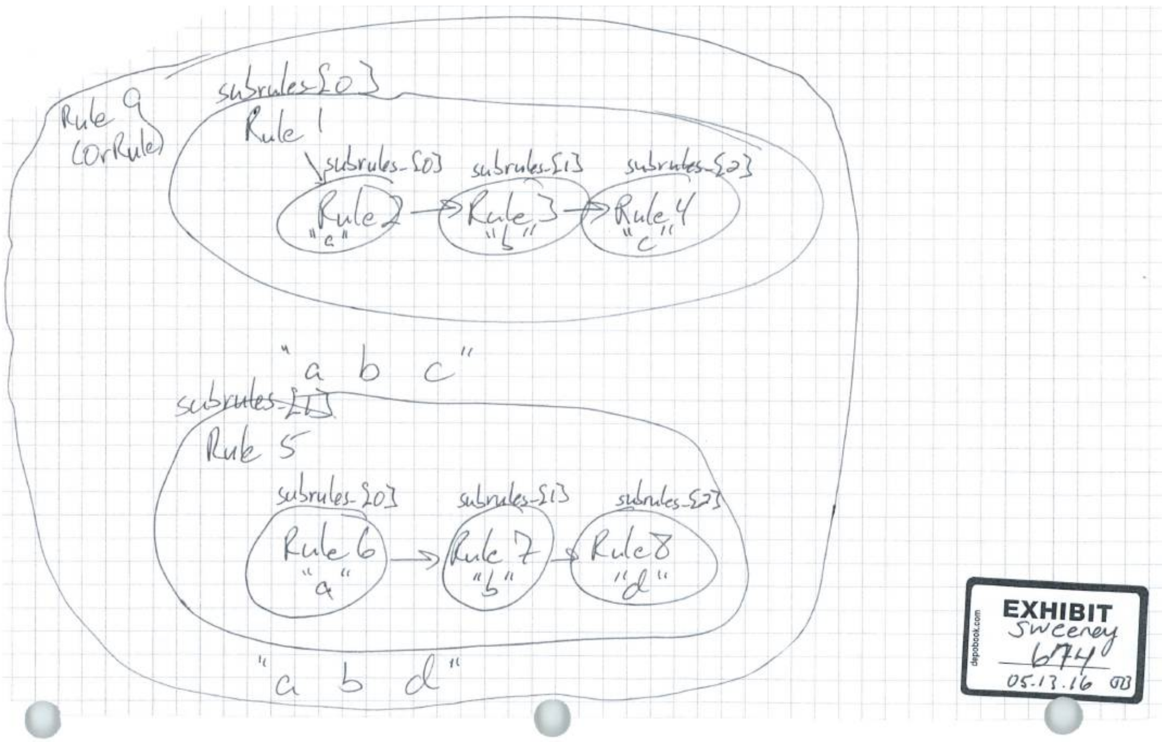
182. Specifically, the EOS CLI parser, while parsing CLI tokens performs the following steps: On line [CliRule.py : 1114], OrRule.inhale() calls OrRule.tryToInhale() [CliRule.py : 1003-1039]. For each remaining subrule "child" in its context.state_, on line [CliRule.py : 1012] this function calls childInhale() with the child as input. childInhale() [CluRule.py : 200-201] calls child.inhale() which invokes the inhale() function of whatever type that child is. By passing the childInhale() function with the child object as input, which in turn calls child.inhale, this traverses the command parse tree recursively. *See* Paragraphs 99-116, which are hereby incorporated by reference.

E. The Accused Products Include the Limitations of Claim 4

183. In my opinion, each of the Accused Products practices the following limitation [4.0] "The method of claim 3, wherein the issuing step includes issuing the prescribed command based on a corresponding command key specified for the matching token within the identified one element."

184. When traversing the command parse tree as described for claim 1, the EOS CLI issues a prescribed command. See Exhibit 4 (list of prescribed commands). It does so based on a corresponding command key—the value function or "action" of the matching rule. *See* Paragraphs 99-116, which are hereby incorporated by reference. As discussed above, when the EOS CLI parser is traversing the command parse tree, it looks at each rule or subrule and tests

whether the token from the user matches the token in the rule. In the case of Mr. Sweeney's example from his deposition, when the user types "a b c" Rule 1 will match the CLI input from the user to the value function for Rule 1.



F. The Accused Products Include the Limitations of Claim 5

185. In my opinion, each of the Accused Products practices the following limitation [5.0] "The method of claim 4, wherein the issuing step further includes accessing a prescribed translator configured for converting the generic command according to the corresponding command format into the prescribed command based on the corresponding command key."

186. As described above, the EOS CLI parser translates CLI commands until a value function is called, whether by an exact match or a non-ambiguous partial command entered by the user. *See* Paragraphs 82-111 (describing command parsing and partial command parsing). The value function, in turn, will either change state in Sysdb or otherwise activate the relevant

function in a EOS agent. *See* Paragraphs 112-120(describing Sysdb). As shown in the list of generic commands in Exhibit 4, the generic commands are translated from the generic command format to the prescribed command format by the invoked value function. For example: generic command "show kernel ip counters [vrf <vrfName>]" is translated in Ira/CliPlugin/IraKernelIpCli.py to "netstat -s -A inet." As another example, generic command "show platform arad arp" is translated to the prescribed command "/usr/bin/SandL3UnicastCliUtil --arp=<switch list, comma-separated> --platform=arad" in by the SandL3Unicast agent implemented in SandL3Unicast/CliPlugin/SandL3UnicastCli.py. This translation is based on the corresponding command key, as described above for claim 4.

G. <u>The Accused Products Include the Limitations of Claim 6</u>

187. In my opinion, each of the Accused Products practices the following limitation [6.0] "The method of claim 5, wherein the validating step including validating at least a portion of the generic command by identifying the one element having the best match relative to the portion of the generic command."

188. The EOS CLI parser will accept and successfully parse partial commands, when they are non-ambiguous.

See? Just like IOS, even down to the behavior of truncated commands being accepted, so long as they are not ambiguous. For example, while in configuration mode, entering just `ro` will not work because the CLI interpreter cannot figure out if I mean `route-map` or `router`:

```
Arista-1(config)#ro
% Ambiguous command
```

I can, however, find out what commands are available in one of two ways. First, I can hit question mark. This will give me a list of available commands that match what I've entered so far:

```
Arista-1(config)#ro?
route-map  router
```

I can also hit the Tab key, at which point the switch will respond with the longest match based on what I've typed so far:

```
Arista-1(config)#ro<TAB>
Arista-1(config)#route
```

Note that I did not type the word `route`; the switch inserted that via *autocompletion* when I hit Tab. At this point I decided that it's the `router` command I was looking for, so I added the `r`, and then hit question mark. The switch then recognized that `router` is a command, and listed the possible associated keywords:

```
Arista-1(config)#router ?
  bgp   Border Gateway Protocol
  ospf  Open Shortest Path First (OSPF)
```

I chose one of these protocols, after which the switch put me into protocol specific mode, and altered the command line to show where I was:

```
Arista-1(config)#router ospf 100
Arista-1(config-router-ospf)#
```

As with IOS, typing `exit` (or its nonambiguous abbreviation) got me out of the current level and popped me back up one level:

```
Arista-1(config-router-ospf)#ex
Arista-1(config)#
```

Arista Warrior at 60-61.

189. As described above, this functionality is implemented in the EOS CLI parser using getCompletions() method for the partial token that was entered by the user. *See* Paragraphs 78-111 (describing parsing of partial tokens by EOS CLI.).

H. <u>The Accused Products Include the Limitations of Claim 10</u>

1. The Accused Products Meet the Limitations of the Preamble to Claim 10

190. In my opinion, each of the Accused Products practices the following limitation [10.0] " A system configured for executing a plurality of management programs according to respective command formats, the system comprising:"

191. See analysis for claim element [1.0], which I incorporate by reference.

2. The Accused Products Meet the Limitations of Claim 10.1

192. In my opinion, each of the Accused Products practices the following limitation [10.1] " a parser having a command parse tree configured for validating a generic command received from a user"

193. See analysis for claim element [1.1], which I incorporate by reference.

3. The Accused Products Meet the Limitations of Claim 10.2

194. In my opinion, each of the Accused Products practices the following limitation [10.2] " the command parse tree configured for specifying valid generic commands relative to a prescribed generic command format"

195. See analysis for claim element [1.2], which I incorporate by reference.

4. The Accused Products Meet the Limitations of Claim 10.3

196. In my opinion, each of the Accused Products practices the following limitation [10.3] " and having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value"

197. See analysis for claim element [1.3], which I incorporate by reference.

5. The Accused Products Meet the Limitations of Claim 10.4

198. In my opinion, each of the Accused Products practices the following limitation [10.4] "the parser identifying one of the elements as a best match relative to the generic command value."

199. See analysis for claim element [1.4], which I incorporate by reference.

6. The Accused Products Meet the Limitations of Claim 10.5

200. In my opinion, each of the Accused Products practices the following limitation [10.5] "a plurality of translators configured for issuing commands for the management programs according to respective command formats."

201. See analysis for claim element [5], which I incorporate by reference. As described therein, the agents that are invoked by each value function for the generic commands comprise a plurality of translators which issues prescribed commands according to respective command formats. *See* Exhibit 4.

7. The Accused Products Meet the Limitations of Claim 10.6

202. In my opinion, each of the Accused Products practices the following limitation [10.6] "the parser outputting a prescribed command to a selected one of the translators based on the identified one element.."

203. See analysis for claim element [1.5], which I incorporate by reference.

I. The Accused Products Include the Limitations of Claim 11

1. The Accused Products Meet the Limitations of Claim 11.0

204. In my opinion, each of the Accused Products practices the following limitation [11.0] "The system of claim 10, wherein the parser further comprises a command word translation table configured for storing for each prescribed command word a corresponding token for identification of a matching token."

205. See analysis for claim element [2.1], which I incorporate by reference.

2. The Accused Products Meet the Limitations of Claim 11.1

206. In my opinion, each of the Accused Products practices the following limitation [11.1] "the parser configured for determining a presence of the matching token within the command parse tree for each input command word."

207. See analysis for claim element [2.2], which I incorporate by reference.

J. The Accused Products Include the Limitations of Claim 12

208. In my opinion, each of the Accused Products practices the following limitation [12.0] "The system of claim 11, wherein the parser recursively traverses the command parse tree

based on an order of the input command words for identification of the matching token within the identified one element."

209. See analysis for claim element [3.0], which I incorporate by reference.

K. The Accused Products Include the Limitations of Claim 13

210. In my opinion, each of the Accused Products practices the following limitation [12.0] "The system of claim 12, wherein the parser validates at least a portion of the generic command by identifying the one element having the best match relative to the portion of the generic command."

211. See analysis for claim element [6.0], which I incorporate by reference.

L. The Accused Products Include the Limitations of Claim 14

1. The Accused Products Meet the Limitations of Claim 14.0

212. In my opinion, each of the Accused Products practices the following limitation [14.0] "A computer readable medium having stored thereon sequences of instructions for executing a plurality of management programs according to respective command formats, the sequences of instructions including instructions for performing the steps of."

213. See analysis for claim element [1.0], which I incorporate by reference.

214. In addition, EOS comprises a computer readable medium having stored thereon sequences of instructions, including that it is software (sequences of instructions) that reside in memory, which comprises a computer readable medium. See, e.g., Arista 7500 Series Data Center Switch Data Sheet;

2. The Accused Products Meet the Limitations of Claim 14.1

215. In my opinion, each of the Accused Products practices the following limitation [14.1] "receiving a generic command from the user."

216. See analysis for claim element [1.1], which I incorporate by reference.

3. The Accused Products Meet the Limitations of Claim 14.2

217. In my opinion, each of the Accused Products practices the following limitation [14.2] "validating the generic command based on a command parse tree that specifies valid generic commands relative to a prescribed generic command format."

218. See analysis for claim element [1.2], which I incorporate by reference.

4. The Accused Products Meet the Limitations of Claim 14.3

219. In my opinion, each of the Accused Products practices the following limitation [14.3] "the command parse tree having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value."

220. See analysis for claim element [1.3], which I incorporate by reference.

5. The Accused Products Meet the Limitations of Claim 14.4

221. In my opinion, each of the Accused Products practices the following limitation [14.4] "the validating step including identifying one of the elements as a best match relative to the generic command."

222. See analysis for claim element [1.4], which I incorporate by reference.

6. The Accused Products Meet the Limitations of Claim 14.5

223. In my opinion, each of the Accused Products practices the following limitation [14.5] "issuing a prescribed command of a selected one of the management programs according to the corresponding command format, based on the identified one element."

224. See analysis for claim element [1.5], which I incorporate by reference.

M. The Accused Products Include the Limitations of Claim 15

1. The Accused Products Meet the Limitations of Claim 15.0

225. In my opinion, each of the Accused Products practices the following limitation [15.0] "The medium of claim 14, wherein the generic command includes at least one input command word."

226. See analysis for claim element [2.0], which I incorporate by reference.

2. The Accused Products Meet the Limitations of Claim 15.1

227. In my opinion, each of the Accused Products practices the following limitation [15.1] "the validating step including comparing each input command word to a command word translation table, configured for storing for each prescribed command word a corresponding token, for identification of a matching token; and."

228. See analysis for claim element [2.1], which I incorporate by reference.

3. The Accused Products Meet the Limitations of Claim 15.2

229. In my opinion, each of the Accused Products practices the following limitation [15.2] "determining a presence of the matching token within the command parse tree for each input command word."

230. See analysis for claim element [2.2], which I incorporate by reference.

N. The Accused Products Include the Limitations of Claim 16

231. In my opinion, each of the Accused Products practices the following limitation [16.0] "The medium of claim 15, wherein the determining step includes recursively traversing the command parse tree based on an order of the input command words for identification of the matching token within the identified one element."

232. See analysis for claim element [3.0], which I incorporate by reference.

O. The Accused Products Include the Limitations of Claim 17

233. In my opinion, each of the Accused Products practices the following limitation [17.0] "The medium of claim 16, wherein the issuing step includes issuing the prescribed command based on a corresponding command key specified for the matching token within the identified one element."

234. See analysis for claim element [4.0], which I incorporate by reference.

P. The Accused Products Include the Limitations of Claim 18

235. In my opinion, each of the Accused Products practices the following limitation [18.0] "The medium of claim 17, wherein the issuing step further includes accessing a prescribed translator configured for converting the generic command according to the corresponding command format into the prescribed command based on the corresponding command key."

236. See analysis for claim element [5.0], which I incorporate by reference.

Q. The Accused Products Include the Limitations of Claim 19

1. The Accused Products Meet the Limitations of Claim 19.0

237. In my opinion, each of the Accused Products practices the following limitation [19.0] "The medium of claim 18, wherein the validating step including validating at least a portion of the generic command by identifying the one element having the best match relative to the portion of the generic command."

238. See analysis for claim element [6.0], which I incorporate by reference.

2. The Accused Products Meet the Limitations of Claim 19.1

239. In my opinion, each of the Accused Products practices the following limitation [19.1] "the issuing step including issuing the prescribed command based on the identified one element corresponding to the portion of the generic command."

240. See analysis for claim element [6.1], which I incorporate by reference.

R. The Accused Products Include the Limitations of Claim 23

241. Because Cisco and Arista break down their proposed functions and structures differently, not all of the analysis here fits neatly into the same structured sub-limitations. Nevertheless, I provide infringement opinions based on both Cisco's and Arista's proposed structures.

1. The Accused Products Meet the Limitations of Claim 23.0

242. In my opinion, each of the Accused Products practices the following limitation [23.0] "A system configured for executing a plurality of management programs according to respective command formats, the system comprising."

243. See analysis for claim element [1.0], which I incorporate by reference.

2. The Accused Products Meet the Limitations of Claim 23.1

244. In my opinion, each of the Accused Products practices the following limitation [23.1] "means for validating a generic command received from a user."

245. See analysis for claim element [1.0], which I incorporate by reference. In addition, the Arista EOS products include the same corresponding structure as identified in the '526 patent, including a command parse tree that includes command keys and command action values, as described above. To the extent not literally present, the EOS CLI parser is an equivalent structure to the disclosed corresponding means for validating a generic command received from a user described in the '526 specification, including Figure 3 of the '526 patent. Both include a command parse tree, command keys and command action values, as described above. Both perform the same function: they take generic commands and validate them in order to issue a corresponding prescribed command. Both perform the function in the same way: both use command parse trees, command action values and command keys, as described above, to implement the generic command validation feature. Both provide the same result: a generic

command is validated and a prescribed command is issued after being translated from the generic command format.

3. The Accused Products Meet the Limitations of Claim 23.2

246. In my opinion, each of the Accused Products practices the following limitation [23.2] "the validating means configured for specifying valid generic commands relative to a prescribed generic command format."

247. See analysis for claim element [1.2], which I incorporate by reference.

4. The Accused Products Meet the Limitations of Claim 23.3

248. In my opinion, each of the Accused Products practices the following limitation [23.3] "and having elements each specifying at least one corresponding generic command component and a corresponding at least one command action value."

249. See analysis for claim element [1.3], which I incorporate by reference.

5. The Accused Products Meet the Limitations of Claim 23.4

250. In my opinion, each of the Accused Products practices the following limitation [23.4] "the validating means identifying one of the elements as a best match relative to the generic command; and."

251. See analysis for claim element [1.4], which I incorporate by reference.

6. The Accused Products Meet the Limitations of Claim 23.5

252. In my opinion, each of the Accused Products practices the following limitation [23.5] "a plurality of translators configured for issuing commands for the management programs according to respective command formats."

253. See analysis for claim element [10.5], which I incorporate by reference.

7. **The Accused Products Meet the Limitations of Claim 23.6**

254. In my opinion, each of the Accused Products practices the following limitation [23.6] "the validating means outputting a prescribed command to a selected one of the translators based on the identified one element."

255. See analysis for claim element [10.6], which I incorporate by reference.

256. My infringement opinion remains the same under either Cisco's proposed construction or Arista's proposed construction. Cisco and Arista propose functions that parrot the claim limitations of claim 23. These are described above in the paragraphs for Claims 23.1 to 23.6.

257. Cisco proposes a structure of "Parser 14 in Figure 2, which includes the command word translation table 20 and the command parse tree 22, as described in 3:36-61, and equivalents." Figure 2 describes an embodiment of the parser and depicts a command parse tree. The section above for Claim 1.2 shows how the accused EOS CLI command parse tree uses the structure disclosed in Figure 2 (see in particular paragraphs 49 and 49). The command word translation table 20 is found in the Accused Products, as described in Claim 2.1. The command parse tree is found in the Accused Products, as described in Claim 1.2. This is described in the sections for Claims 1.2 to 1.4 above.

258. My infringement opinion remains the same under Arista's proposed structure as well. In particular, Arista proposes the following for its structure: "A processor executing a parser, and a corresponding memory storing a command parse tree, wherein the parser executes the algorithm of Figure 3, and wherein (1) each node of the command parse tree specifies one token and a corresponding command key; (2) the top-level nodes of the command parse tree represent all possible valid first words in the input command, second-level nodes represent all possible valid second words for each valid first word in the input command, and so on;"

259. As explained in the section describing the Preamble to Claim 1, Arista's system includes a processor that executes a parser, and as explained in the section for Claim 1.2, the command parse tree is stored in memory in the form of rules. The sections for Claims 1.2 to 1.4 explain how EOS CLI implement the steps of the algorithm listed in Figure 3.

260. The section for Claim 1.3 explains that each node of the command parse tree specifies one token and a corresponding command key. For example, see the explanation in paragraph 51, and the paragraphs cited therein.

261. The section for Claim 1.2 shows the structure of a sample command parse tree, where first words are shown at the top level and second words are shown at the next level. An example of a command parse tree is shown in paragraph 49.

VII. INDIRECT INFRINGEMENT

262. I have discussed above the direct infringement of the '526 patent in the previous section of this report. In this section, I discuss my conclusion that Arista not only directly infringes, but also indirectly infringes, via both induced and contributory infringement, the asserted claims of the '526 patent through its sale, distribution, and promotion of the Accused Products.

263. Although this indirect infringement section is separate from the direct infringement section, this discussion should be understood in the context of the element by element analysis I provided above.

A. Induced Infringement

264. It is my opinion that Arista induces infringement of the '526 patent by selling the Accused Products and actively encouraging and instructing those customers to infringe the '526 patent, with knowledge of the '526 patent, and with the specific intent to cause infringement.

1. **Arista Customers Directly Infringe**

265. First, I understand that customers of the Accused Products infringe the asserted system and program storage device claims by using the products and the CLI interface, including the generic commands and management programs described above.

266. Second, customers of the Accused Products infringe the asserted method claims in addition to the asserted system and program storage device claims by using the Accused Products in the manner set forth above in my infringement analysis. The features accused of infringing the '526 patent are enabled and available by default. Arista includes in its command reference manual the generic commands identified above, instructing its users to operate the Accused '526 Products in an infringing fashion for the reasons I identified above. *See generally* CSI-CLI-00007473, CSI-CLI-00007244, CSI-CLI-00006858, CSI-CLI-00007841, CSI-CLI-00010517, CSI-CLI-00008985, CSI-CLI-00014141, CSI-CLI-00011973, CSI-CLI-00018146, CSI-CLI-00000084, CSI-CLI-00004616, CSI-CLI-00020575, CSI-CLI-00002332, CSI-CLI-00016001.

2. **Arista Had Knowledge of the '526 Patent**

267. Arista had knowledge of the '526 patent at least as early as the filing of Cisco's Complaint.

3. **Arista Encourages and Instructs its Customers to Infringe**

268. Arista encourages customers of the Accused Products to infringe the '526 patent's system and program storage device claims by encouraging them to use the products, regardless of configuration. Arista further encourages customers of the Accused Products to infringe the asserted method claims in addition the asserted system and program storage device claims by encouraging them to employ the accused features. *See generally* CSI-CLI-00007473, CSI-CLI-

00007244, CSI-CLI-00006858, CSI-CLI-00007841, CSI-CLI-00010517, CSI-CLI-00008985, CSI-CLI-00014141, CSI-CLI-00011973, CSI-CLI-00018146, CSI-CLI-00000084, CSI-CLI-00004616, CSI-CLI-00020575, CSI-CLI-00002332, CSI-CLI-00016001.

269. Arista includes in its command reference manual the generic commands identified above, instructing and encouraging its users to operate the Accused Products in an infringing fashion for the reasons I identified above. *See generally* CSI-CLI-00007473,CSI-CLI-00007244, CSI-CLI-00006858, CSI-CLI-00007841, CSI-CLI-00010517, CSI-CLI-00008985, CSI-CLI-00014141, CSI-CLI-00011973, CSI-CLI-00018146, CSI-CLI-00000084, CSI-CLI-00004616, CSI-CLI-00020575, CSI-CLI-00002332, CSI-CLI-00016001.

270. In my opinion, the evidence establishes that there is direct infringement of the '526 patent by Arista's customers, that Arista knew about the '526 patent, that Arista knew that its customers' usage of the Accused Products infringed the '526 patent, but yet Arista instructed and encouraged its customers to take actions that directly infringe the '526 patent.

B. <u>Contributory Infringement</u>

271. Arista contributorily infringes the '526 patent by selling the Accused Products to its customers. I understand that contributory infringement requires a direct infringement by a Arista customer, which I have described above in connection with my inducement analysis. I also understand that contributory infringement requires knowledge of the patents, which I have likewise described above in connection with my inducement analysis. I will address the remaining requirements for contributory infringement here.

1. Arista Knew that the Accused Products were Especially Made or Especially Adapted for use in the Infringement of the '526 Patent

272. As mentioned above, Arista was aware of the '526 patent and Cisco's infringement allegations against the Accused Products after the complaint was filed in this action. Based on those facts, Arista knew that the accused functionality would perform in an infringing manner after the complaint was filed. Arista therefore designed its post-Complaint accused functionality to infringe the asserted claims of the '526 patent.

2. The Accused Products have no Substantial Non-Infringing Uses

273. In my opinion, without the accused functionality, the Accused Products have no substantial non-infringing uses. EOS is designed as an "extensible" operating system, with the "agent" functionality designed as a core element of the operating system. Removing the agent structure and the ability to use generic commands to access these management programs would materially change the nature of EOS and the Accused Products. *See* ANI-ITC-944_945-0779972) at 4 ("One of the big parts of our integration work is to design the chassis and the board in which all the components we select will live and work together."); ANI-ITC-944_945-0771285; ANI-ITC-944_945-0771293; ANI-ITC-944_945-0718840; ANI-ITC-944_945-0779967; ANI-ITC-944_945-1368604; ANI-ITC-944_945-1369644. *see also*, ANI-ITC-944_945-1624373.

274. Based on this evidence, I conclude that Arista contributorily infringes the '526 patent by selling the Accused Products.

[redacted]

I certify under penalty of perjury that the foregoing is true and correct.

Date: June 3, 2015.

Kevin Jeffay, Ph.D.

EXHIBIT 1

CURRICULUM VITAE

Kevin Jeffay

University of North Carolina at Chapel Hill
Department of Computer Science
Chapel Hill, NC 27599-3175
(919) 590-6238
jeffay@cs.unc.edu

Education

Ph.D. Computer Science, University of Washington.
Honors: IBM Graduate Fellowship.

M.Sc. Computer Science, University of Toronto.
Honors: University of Toronto Open Fellowship.

B.S. Mathematics with Highest Distinction, University of Illinois at Urbana-Champaign.
Honors: Phi Beta Kappa, James Scholar.

Academic Experience

Chairman, Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC, 2014–present.

Gillian T. Cell Distinguished Professor in Computer Science, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, 2008–present.

S. S. Jones Distinguished Professor in Computer Science, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, 2001–2008.

S. S. Jones Distinguished Term Associate Professor of Computer Science, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, 2000.

Associate Professor, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, 1996–2000.

Visiting Professor, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA, 1994.

Assistant Professor, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, 1989–1995.

Honors and Awards

Alumni Achievement Award, University of Washington, Department of Computer Science and Engineering, June 2015.

Excellence in Teaching Award, Computer Science Student Association, University of North Carolina at Chapel Hill, Department of Computer Science, May 2011.

Favorite Faculty Award, an award given by the Computer Science majors of the 2010 graduating class, May 2010.

Carolina Women's Center Women’s Advocacy Award, University of North Carolina at Chapel Hill, Chapel Hill, NC, March 2010.

Gillian T. Cell Distinguished Professor in Computer Science, University of North Carolina at Chapel Hill, College of Arts and Sciences, Chapel Hill, NC, June 2008.

Outstanding Teaching Award, an award given by the Computer Science majors of the 2008 graduating class, May 2008.

Favorite Faculty Award, an award given by the Computer Science majors of the 2004 graduating class, May 2004.

Edward Kidder Graham Outstanding Faculty Award, an award given by the Class of 2004 and the General Alumni Association, April 2004.

Edward Kidder Graham Advisor of the Year Award, an award given by the Class of 2004 and the General Alumni Association, April 2004.

IEEE Service Award, Service to the IEEE Technical Committee on Real-Time Systems, December 2001.

Outstanding Teaching Award, Computer Science Student Association, University of North Carolina at Chapel Hill, Department of Computer Science, May 2000.

S. Shepard Jones Distinguished Term Professorship, University of North Carolina at Chapel Hill, College of Arts and Sciences, Chapel Hill, NC, March 1999.

Award Papers:

ACM SIGCOMM 2000, 2003.

IEEE International Symposium on High Assurance Systems Engineering 1999, 2000.

IEEE Real-Time Technology and Applications Symposium, 1995.

ACM International Conference on Multimedia, 1994.

Computer Networks and ISDN Systems, 1994.

International Workshop on Network and Operating System Support for Digital Audio and Video, 1991, 1992, 1993, 1997.

Entities From Whom Compensation Has Been Received (Within Last Five Years)

Cisco Systems, Inc.	Alcatel-Lucent USA
Global Tel*Link	Vonage
Samsung Electronics Co.	Xerox
LG Electronics U.S.A., Inc.	Worldvoice
T-Mobile USA, Inc.	Comcast
HTC Corp.	Verizon Communications Inc.
AT&T Services, Inc.	Google
Citrix Systems, Inc.	Yahoo!
Hewlett-Packard	Progressive Insurance
Seachange International, Inc.	Hyatt Corp.
Microsoft, Inc.	Everbridge
Lexmark	Sony Corp.
A10 Networks	Akamai
Motorola Mobility	Time Warner Cable
Amway	YouTube
Hulu	St. Jude Medical

Expert Testimony (Within Last Five Years)

In re: Post Grant Review of U.S. Patent No. 8,929,525.

Provided deposition testimony on behalf of Petitioner Global Tel*Link, Inc.

In the Matter of: Certain Network Devices, Related Software and Components Thereof (II), United States International Trade Commission, Investigation No. 337-TA-945.

Provided deposition and hearing testimony on behalf of complainant Cisco Systems Inc.

In the Matter of: Certain Network Devices, Related Software and Components Thereof, United States International Trade Commission, Investigation No. 337-TA-944.

Provided deposition and hearing testimony on behalf of complainant Cisco Systems Inc.

In re: Inter Partes Review of U.S. Patent Nos. 8, 045,952 and 8,050,652.

Provided deposition testimony on behalf of Petitioner Samsung Electronics Co., Ltd., Samsung Electronics Co., Ltd., Samsung Electronics America, Inc., and Samsung Telecommunications America, LLC.

In the Matter of: Certain Digital Media devices, Including Televisions, Blu-Ray Disc Players, Home Theater Systems, Tablets and Mobile Phones, Components Thereof and Associated Software, United States International Trade Commission, Investigation No. 337-TA-882.

Provided deposition and hearing testimony on behalf of respondents Samsung Electronics Co., Ltd., Samsung Electronics America, Inc., and Samsung Telecommunications America, LLC, LG electronics, Inc., LG Electronics U.S.A., Inc., and LG Electronics MobileComm U.S.A., Inc., Panasonic Corporation and Panasonic Corporation of America, and Toshiba Corporation and Toshiba America Information Systems, Inc., and intervenor Google.

Apple Inc. v. Samsung Electronics Co., Ltd., Samsung Electronics America, Inc., Samsung Telecommunications America, United States District Court for the Northern District of Californian, San Jose Division, Case No. 12-cv-00630-LHK.

Provided deposition and trial testimony on behalf of Samsung defendants.

LinkSmart Wireless Technologies, LLC v. T-Mobile USA, Inc., et al., United States District Court for the Central District of California, Santa Ana Division, Case No. SACV12-00522-AG.

Provided deposition testimony on behalf of defendants Cisco Systems Inc., T-Mobile USA, Inc., and related defendants.

MobileMedia Ideas LLC v. HTC Corp. and HTC America Inc., United States District Court for the Eastern District of Texas, Marshall Division, Case No. 2:10-CV-112-JRG.

Provided deposition testimony on behalf of defendants HTC Corp. and HTC America Inc.

Richard A. Williamson v. Verizon Communications Inc., et al., United States District Court for the Southern District of New York, Case No. 1:11-cv-04948 (LTS(HBP), *Richard A. Williamson v. AT&T Operations Inc. and AT&T Services, Inc.*, United States District Court for the Southern District of New York, Case No. 1:13-cv-00645 (LTS)(HBP).

Provided deposition testimony on behalf of defendants Verizon Communications Inc., *et al.* and AT&T Operations, Inc., and AT&T Services, Inc.

Pixion, Inc., v. Citrix Systems, Inc., Citrix Online, LLC, United States District Court for the Northern District of California, San Francisco Division, Case No. 09-cv-03496-SI.

Provided deposition testimony on behalf of defendants Citrix Systems, Inc., and Citrix Online, LLC.

Nomadix, Inc., v. Hewlett-Packard Company, et al., United States District Court for the Central District of California Western Division, Case No. 09-cv-8441-DDP (VBK).

Provided deposition testimony on behalf of defendants Hewlett-Packard Company, Wayport, Inc., iBAHN Corporation, iBAHN General Holdings Corporation, Aruba Networks, Inc., Superclick, Inc., and Superclick Networks, Inc.

nCUBE Corporation (now ARRIS Group, Inc.), v. SeaChange International, Inc., United States District Court for the District of Delaware (C.A. No. 01-011 (LPS)).

Provided deposition and hearing testimony on behalf of defendant SeaChange International, Inc.

Cooper Notification Inc., v. Twitter, Inc., et al., United States District Court for the District of Delaware, (Case No. 09-C-865-JIF).

Provided deposition testimony on behalf of defendant Everbridge Inc.

TIVO Inc., v. AT&T Inc., United States District Court for the Eastern District of Texas, Marshall Division, (Case No. 2:09-CV-0259-DF).

Provided deposition testimony on behalf of intervenor Microsoft Corp.

Extreme Networks, Inc., v. Enterasys Networks, Inc., United States District Court for the Western District

of Wisconsin, (07-C-0229-C).

Provided deposition and trial testimony on behalf of defendant Enterasys Networks.

Alcatel-Lucent USA, Inc. v. Amazon.com, Inc. et al., United States District Court for the Eastern District of Texas, Tyler Division, (Civil Action No. 6:09-cv-422-LED).

Provided deposition testimony on behalf of counter-claim defendant Alcatel-Lucent USA, Inc.

F5 Networks, Inc. v. A10 Networks, Inc., United States District Court for the Western District of Washington, Seattle Division, (Case No. C10-00654 MJP).

Provided deposition testimony on behalf of defendant A10 Networks, Inc.

In the Matter of: Certain Personal Data and Mobile Communications Devices and Related Service, United States International Trade Commission, Investigation No. 337-TA-710.

Provided deposition and hearing testimony on behalf of respondents HTC Corp., HTC America, Inc., and Exedea, Inc.

Bedrock Computer Technologies LLC v. Softlayer Technology, Inc., et al., United States District Court for the Eastern District of Texas, Tyler Division (Case No. 6:09-CV-0029).

Provided deposition and trial testimony on behalf of defendants Google, Inc., and Match.com LLC.

Industry Experience

Consultant, Mälardalen University, Vásteras, Sweden, 2001.

Member, Technical Advisory Board, Ganymede Software Inc., Research Triangle Park, NC, 1996–1999.

Consultant and Author, INTEROP Graduate Institute, SOFTBANK Inc., Foster City, CA, 1996–1997.

Consultant, National Science Foundation, Arlington, VA, 1995–present.

Consultant, Monterey Technologies Inc., Cary, NC, 1994–1996.

Consultant, Hewlett-Packard Inc., Ink Jet Components Division, Corvallis, OR, 1993–1994.

Visiting Researcher, IBM T.J. Watson Research Center, Computer Systems Principles Group, Yorktown Heights, NY, 1986.

Consultant, Boeing Aerospace, Kent, WA, 1985–1986.

Software Engineer, R.R. Donnelley & Sons Company Inc., Technical Center, Chicago, IL, 1984.

Computer Programmer, US Army Corps of Engineers Construction Engineering Research Lab, Champaign, IL, 1981–1982.

Community Service

Consultant (unpaid), Children's Museum About the World (now called *Exploris*), Raleigh, NC, 1995–1996.

Professional Activities

Editor

Associate Editor, *Real-Time Systems*, Kluwer Academic Publishers, The Netherlands, 2003-present.

Editor in Chief, *Multimedia Systems*, ACM/Springer-Verlag, Heidelberg, Germany, 2000–2001.

Editorial Board

Journal of Multimedia Tools and Applications, Kluwer Academic Publishers, 1994–1999.

Guest Editor

Computer Communications, special issue on system support for multimedia computing, Volume 18, Number 10, October 1995.

Executive Committees

ACM/SIGCOMM Internet Measurement Conference Steering Committee, 2005-2009.

College of Reviewers, Canada Research Chairs Program, 2004-present.

Statistical and Applied Mathematical Sciences Institute (SAMSI) program on Network Modeling for the Internet, 2002-2004.

IEEE Technical Committee on Real-Time Systems, 2000-present.

ACM Special Interest Group on Multimedia, 2000-2002.

Other Committees

IEEE Distinguished Visitors Program, 2004–2007.

Conference Program Chair

11th International Workshop on Quality-of-Service (Co-Chair), Monterey, CA, June 2003.

Sixth IEEE Symposium on Computers and Communications, Hammamet, Tunisia, July 2001.

21st IEEE Real-Time Systems Symposium, Orlando, FL, November 2000.

Tenth International Workshop on Network and Operating System Support for Digital Audio and Video, Chapel Hill, NC, June 2000.

Seventh ACM International Conference on Multimedia (Co-Chair), Orlando, FL, November 1999.

SPIE/ACM Multimedia Computing and Networking 1999 (Co-Chair), San Jose, CA, January 1999.

Sixth ACM International Conference on Multimedia (Associate Chair), Bristol, UK, September 1998.

IEEE International Conference on Multimedia Computing Systems (Associate Chair), Austin, TX, June 1998.

SPIE/ACM Multimedia Computing and Networking 1998 (Co-Chair), San Jose, CA, January 1998.

SPIE/ACM Multimedia Computing and Networking 1997 (Associate Chair), San Jose, CA, February 1997.

IEEE Workshop on Resource Allocation Problems in Multimedia Systems, Washington D.C., December 1996.

Second IEEE Real-Time Technology and Applications Symposium, Boston, MA, June 1996.

Conference General Chair

ACM SIGCOMM Internet Measurement Conference, San Diego, CA, October 2007.

22nd IEEE Real-Time Systems Symposium, London, UK, December 2001.

Tenth International Workshop on Network and Operating System Support for Digital Audio and Video, Chapel Hill, NC, June 2000.

Third IEEE Real-Time Technology and Applications Symposium, Montreal, Canada, June 1997.

IEEE Workshop on Resource Allocation Problems in Multimedia Systems, Washington D.C., December 1996.

Member of Conference Program Committee

21st IEEE International Conference on Network Protocols, Goettingen, Germany, October 2013.

18th IEEE International Conference on Network Protocols, Kyoto, Japan, October 2010.

ACM SIGCOMM 2010, New Delhi, India, August 2010.

17th IEEE International Conference on Network Protocols, Princeton, NJ, October 2009.

9th Passive and Active Measurement Conference 2008, Cleveland, OH, April 2008.

ACM SIGMETRICS 2008, Annapolis, MA, June 2008.

IEEE INFOCOM 2008, Phoenix, AZ, April 2008.

6th ACM SIGCOMM Workshop on Hot Topics in Networks, Atlanta, GA, November 2007.

15th IEEE International Conference on Network Protocols, Beijing, China, October 2007.

5th IEEE Workshop on Embedded Systems for Real-Time Multimedia, Salzburg, Austria, October 2007.

Workshop on Experimental Computer Science, ACM FCRC, San Diego, CA, June 2007.

15th International Workshop on Quality-of-Service, Chicago, IL, June 2007.

17th International Workshop on Network and Operating System Support for Digital Audio and Video, Urbana-Champaign, IL, June 2007.

10th IEEE Global Internet Symposium 2007, Anchorage, AK, May 2007.

ACM SIGMETRICS 2007, San Diego, CA, June 2007.

14th IEEE International Conference on Network Protocols, Santa Barbara, CA, October 2006.

ACM Internet Measurement Conference 2006, Rio de Janeiro, Brazil, October 2006.

Second ACM SIGCOMM Conference on Future Networking Technologies, Lisbon, Portugal, December 2006.

16th International Workshop on Network and Operating System Support for Digital Audio and Video, Newport, RI, June 2006.

IEEE Workshop on Research Directions for Security and Networking in Critical Real-Time and Embedded Systems, San Jose, CA, April 2006.

IEEE INFOCOM 2006, Barcelona, Spain, April 2006.

13th IEEE International Conference on Network Protocols, Boston, MA, November 2005.

15th International Workshop on Network and Operating System Support for Digital Audio and Video, Skamania, WA, June 2005.

IEEE INFOCOM 2005, Miami, FL, March 2005.

SPIE/ACM Multimedia Computing and Networking 2005, San Jose, CA, January 2005.

ACM Multimedia 2004, New York, NY, October 2004.

ACM Internet Measurement Conference 2004, Taormina, Italy, October 2004.

16th EUROMICRO Conference on Real-Time Systems, Catania, Italy, June-July 2004.

14th International Workshop on Network and Operating System Support for Digital Audio and Video, Cork, Ireland, June 2004.

12th International Workshop on Quality-of-Service, Montreal, Canada, June 2004.

10th IEEE Real-time and Embedded Technology and Applications Symposium, Toronto, Canada, May 2004.

IEEE INFOCOM 2004, Hong Kong, March 2004.

24nd IEEE Real-Time Systems Symposium, Cancun, Mexico, December 2003.

19th ACM Symposium on Operating Systems Principles, Lake George, New York, October 2003.

15th EUROMICRO Conference on Real-Time Systems, Porto, Portugal, July 2003.

Ninth IEEE Real-Time and Embedded Technology and Applications Symposium, Toronto, CA, May 2003.

SPIE/ACM Multimedia Computing and Networking 2003, San Jose, CA, January 2003.

23nd IEEE Real-Time Systems Symposium, Austin, TX, November 2002.

10th International Conference on Network Protocols, Paris, France, November 2002.

Second Workshop on Embedded Software, Grenoble, France, October 2002.

22nd IFIP International Symposium on Computer Performance Modeling, Measurement and Evaluation (Performance 2002), Rome, Italy, September 2002.

Eighth IEEE Real-Time and Embedded Technology and Applications Symposium, San Jose, CA, September 2002.

10th International Workshop on Quality-of-Service, Miami, FL, June 2002.

Seventh Global Internet Symposium (held in conjunction with Globecom 2002), Taipei, Taiwan,

November 2002.

14th EUROMICRO Conference on Real-Time Systems, Vienna, Austria, June 2002.

ACM SIGMETRICS 2002, Marina del Rey, CA, June 2002.

SPIE/ACM Multimedia Computing and Networking 2002, San Jose, CA, January 2002.

22nd IEEE Real-Time Systems Symposium, London, UK, December 2001.

Sixth Global Internet Symposium (held in conjunction with Globecom 2001), San Antonio, TX, November 2001.

27th EUROMICRO Conference, Warsaw, Poland, September 2001.

Sixth IEEE Symposium on Computers and Communications, Hammamet, Tunisia, July 2001.

Seventh IEEE Real-Time Technology and Applications Symposium, Taipei, Taiwan, June 2001.

Ninth IFIP International Workshop on Quality of Service, Karlsruhe, Germany, June 2001.

SPIE/ACM Multimedia Computing and Networking 2001, San Jose, CA, January 2001.

Fifth Global Internet Mini-Conference (held in conjunction with Globecom 2000), San Francisco, CA, November 2000.

21st IEEE Real-Time Systems Symposium, Orlando, FL, December 2000.

Tenth International Workshop on Network and Operating System Support for Digital Audio and Video, Chapel Hill, NC, June 2000.

Sixth IEEE Real-Time Technology and Applications Symposium, Washington, D.C., June 2000.

SPIE/ACM Multimedia Computing and Networking 2000, San Jose, CA, January 2000.

19th IEEE Real-Time Systems Symposium, Phoenix, AZ, December 1999.

Fourth Global Internet Mini-Conference (held in conjunction with Globecom '99), Rio de Janero, Brazil, November 1999.

Fifth International Workshop on Multimedia Information Systems, Palm Springs, CA, October 1999.

Ninth International Workshop on Network and Operating System Support for Digital Audio and Video, Basking Ridge, NJ, June 1999.

IEEE Workshop on QoS Support for Real-Time Internet Applications, Vancouver, Canada, June, 1999.

Second ACM Workshop on Internet Server Performance (held in conjunction with SIGMETRICS '99), Atlanta, GA, May 1999.

Seventh IEEE International Workshop on Parallel and Distributed Real-Time Systems, San Juan, Puerto Rico, April 1999.

SPIE/ACM Multimedia Computing and Networking 1998, San Jose, CA, January 1999.

19th IEEE Real-Time Systems Symposium, Madrid, Spain, December 1998.

Third Global Internet Mini-Conference (held in conjunction with Globecom '98), Sydney, Australia November 1998.

Sixth ACM International Conference on Multimedia, Bristol, UK, September 1998.

Third IEEE/ACM International Workshop on Multimedia Database Management Systems, Dayton, OH, August 1998.

Eighth International Workshop on Network and Operating System Support for Digital Audio and Video, Cambridge, UK, July 1998.

SPIE Interactive Multimedia Services and Equipment, Zurich, Switzerland, May 1998.

IEEE Workshop on Dependable and Real-Time E-Commerce Systems, Denver, CO, June 1998.

Fourth IEEE Real-Time Technology and Applications Symposium, Denver, CO, June 1998.

IEEE International Conference on Multimedia Computing Systems, Austin, TX, June 1998.

SPIE/ACM Multimedia Computing and Networking 1998, San Jose, CA, January 1998.

16th IEEE Symposium on Reliable Distributed Systems, Durham, NC, October 1997.

Third IEEE Real-Time Technology and Applications Symposium, Montreal, Canada, June 1997.

IEEE Real-Time Education Workshop, Montreal, Canada, June 1997.

Seventh International Workshop on Network and Operating System Support for Digital Audio and Video, St. Louis, MO, May 1997.

17th International Conference on Distributed Computing Systems, Distributed Real-Time Systems Track, Performance of Distributed Systems Track, Distributed Multimedia Track, Baltimore, MD, May 1997.

Fifth IFIP International Workshop on Quality of Service, New York, NY, May 1997.

Third International Conference on Computer Science & Informatics, Raleigh-Durham, NC, March 1997.

SPIE/ACM Multimedia Computing and Networking 1997, San Jose, CA, February 1997.

Fourth ACM International Conference on Multimedia, Boston, MA, November 1996.

Second USENIX Symposium on Operating Systems Design and Implementation, Seattle, WA, October/November 1996.

ACM SIGSOFT '96: Fourth Symposium on the Foundations of Software Engineering, San Francisco, CA, October 1996.

Second IEEE International Workshop on Multimedia Database Management Systems, Blue Mountain Lake, NY, August 1996.

Second IEEE Real-Time Technology and Applications Symposium, Boston, MA, June 1996.

Sixth International Workshop on Network and Operating System Support for Digital Audio and Video, Zushi, Japan, April 1996.

First International Workshop on Real-Time Databases: Issues and Applications, Newport Beach, CA, March 1996.

SPIE Multimedia Computing and Networking 1996, San Jose, CA, February 1996.

16th IEEE Real-Time Systems Symposium, Pisa, Italy, December 1995.

Third ACM International Conference on Multimedia, Technical Paper Program, San Francisco, CA, November 1995.

Third ACM International Conference on Multimedia, Video Program, San Francisco, CA, November 1995.

1995 ACM Conference on Organizational Computing Systems, Technical Track, Milpitas, CA, August 1995.

Fifth IEEE Workshop on Hot Topics in Operating Systems, Orcas Island, WA, May 1995.

15th International Conference on Distributed Computing Systems, Distributed Real-Time Systems Track and CSCW track, Vancouver, British Columbia, Canada, May 1995.

Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, April 1995.

IEEE Workshop on the Role of Real-Time in Multimedia, Research Triangle Park, NC, December 1993.

14th IEEE Real-Time Systems Symposium, Research Triangle Park, NC, December 1993.

13th International Conference on Distributed Computing Systems, Real-Time Issues Track, Pittsburgh, PA, May 1993.

12th IEEE Real-Time Systems Symposium, San Antonio, TX, December 1991.

IEEE Conference on Communication Software: Communications for Distributed Applications and Systems, Chapel Hill, NC, April 1991.

Conference Organizing Committees

Finance Chair, IEEE Real-Time Systems Symposium, Miami, FL, December 2005.

Finance Chair, IEEE Real-Time Systems Symposium, Lisbon, Portugal, December 2004.

Finance Chair, IEEE Real-Time Systems Symposium, Cancun, Mexico, December 2003.

Poster Session Chair, ACM Symposium on Operating System Principles, Bolton Landing, NY, October 2003.

Demonstrations Chair, ACM Conference on Computer-Supported Cooperative Work, Research Triangle Park, NC, November 1994.

Wine Steward, ACM Symposium on Operating System Principles, Asheville, NC, December 1993.

Local Arrangements Chair, 14th IEEE Symposium on Real-Time Systems, Durham, NC, December 1993.

Refereed Publications

Technical Papers

Simulating Large-Scale Airborne Networks with ns-3, B. Newton, J. Aikat, K. Jeffay, Proceedings of the 2015 ACM Workshop on ns-3, Barcelona, Spain, May 2015, pages 32-39.

Explicit Topology Control for Airborne Networks, B. Newton, K. Jeffay, J. Aikat, Proceedings of the 22nd IEEE International Conference on Network Protocols, Research Triangle, NC, October 2014, pages 368-373.

The Continued Evolution of Web Traffic, B. Newton, K. Jeffay, J. Aikat, Proceedings of the 21st IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), San Francisco, CA, August 2013, pages 80-89.

Gini in a Bottle: A Case Study of Pareto's Principle in the Wild, A. Blate, K. Jeffay, International Journal of Computer Networks and Communications Security, Vol. 1, No. 1, June 2013, pages 30-39.

Experiment Replication Using ProtoGENI Nodes, D. O'Neill, J. Aikat, K. Jeffay, Proceedings of the Second GENI Research and Educational Experiment Workshop, Salt Lake City, UT, March 2013, pages 9-15.

Towards Traffic Benchmarks for Empirical Networking Research: The Role of Connection Structure in Traffic Workload Modeling, J. Aikat, S. Hasan, K. Jeffay, F.D. Smith, Proceedings of the 20th IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Arlington, VA, August 2012, pages 78-86.

Generating Realistic Synthetic TCP Application Workloads, J. Aikat, K. Jeffay, F.D. Smith, Proceedings, GENI Infrastructure Workshop, Princeton, NJ, June 2010, 8 pages.

Correlations of Size, Rate, and Duration in TCP Connections: The Case Against, C. Park, F. Hernández-Campos, J.S. Marron, K. Jeffay, F.D. Smith, Annals of Applied Statistics, Volume 4, Number 1, May 2010, pages 26-52.

Passive, Streaming Inference of TCP Connection Structure for Network Server Management, J. Terrell, K. Jeffay, F.D. Smith, J. Gogan, J. Keller, IFIP International Workshop on Traffic Monitoring and Analysis, Aachen, Germany, May 2009, in, Lecture Notes in Computer Science, Volume 5537, Springer, Berlin, Germany, pages 42-53.

Exposing Server Performance to Network Managers Through Passive Network Measurements, J. Terrell, K. Jeffay, F.D. Smith, J. Gogan, J. Keller, IEEE Internet Network Management Workshop 2008, Orlando, FL, October 2008, pages 1-6.

Multi-Resolution Anomaly Detection for the Internet, L. Zhang, Z. Zhu, K. Jeffay, J.S. Marron, F.D. Smith, IEEE Workshop on Network Management, Phoenix, AZ, April 2008, 6 pages.

The Effects of Active Queue Management and Explicit Congestion Notification on Web Performance, L. Le, J. Aikat, K. Jeffay, F.D. Smith, IEEE/ACM Transactions on Networking, Volume 15, Number 6, December 2007, pages 1217-1230.

Co-Scheduling Variable Execution Time Requirement Real-time Tasks and Non Real-Time Tasks, A. Singh, K. Jeffay, Proceedings of the 19th Euromicro Conference on Real-Time Systems, Pisa, Italy, July 2007, pages 191-200.

Quantifying the Effects of Recent Protocol Improvements to Standards-Track TCP: Impact on Web Performance, M.C. Weigle, K. Jeffay, F.D. Smith, Computer Communications, Volume 29, Number 15, September 2006, pages 2853-2866.

Tmix: A Tool for Generating Realistic TCP Application Workloads in ns-2, M.C. Weigle, P. Adurthi, F. Hernández-Campos, K. Jeffay, F.D. Smith, ACM Computer Communications Review, Volume 36, Number 3, July 2006, pages 67-76.

A Loss and Queuing-Delay Controller for Router Buffer Management, L. Le, K. Jeffay, F.D. Smith, Proceedings of the 26th IEEE International Conference on Distributed Computing Systems, Lisbon, Portugal, July 2006, 10 pages.

Understanding Patterns of TCP Connection Usage with Statistical Clustering, F. Hernández-Campos, A.B. Nobel, F.D. Smith, K. Jeffay, 13th IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Atlanta, GA, September 2005, pages 35-44.

Delay-Based Early Congestion Detection and Adaptation: Impact on Web Performance, M.C. Weigle, K. Jeffay, F.D. Smith, Computer Communications, Volume 8, Number 8, May 2005, pages 837-850.

Extremal Dependence: Internet Traffic Applications, F. Hernández-Campos, K. Jeffay, C. Park, J.S. Marron, S.I. Resnick, Stochastic Models, Volume 21, Number 1, 2005, pages 1-35.

Generating Realistic TCP Workloads, F. Hernández-Campos, F.D. Smith, K. Jeffay, Proceedings of the Computer Measurement Group's 2004 International Conference, Las Vegas, NV, December 2004, pages 273-284.

Differential Congestion Notification: Taming the Elephants, L. Le, J. Aikat, K. Jeffay, F.D. Smith, Proceedings of the 12th IEEE International Conference on Network Protocols, Berlin, Germany, October 2004, pages 118-128.

Stochastic Models for Generating Synthetic HTTP Source Traffic, J. Cao, W.S. Cleveland, Y. Gao, K. Jeffay, F.D. Smith M.C. Weigle, Proceedings of IEEE INFOCOM 2004, Hong Kong, March 2004, Volume 3, pages 1546-1557.

Variability in TCP Roundtrip Times, J. Aikat, J. Kaur, D. Smith, K. Jeffay, Proceedings of the 2003 ACM SIGCOMM Internet Measurement Conference, Miami Beach, FL, October 2003, pages 279-284.

Tracking the Evolution of Web Traffic: 1995-2003, F. Hernández-Campos, K. Jeffay, F.D. Smith, Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Orlando, FL, October 2003, pages 16-25.

The Effects of Active Queue Management on Web Performance, L. Le, J. Aikat, K. Jeffay, F.D. Smith, Proceedings of ACM SIGCOMM 2003, Karlsruhe, Germany, August 2003, pages 265-276.

Managing Latency and Buffer Requirements in Processing Graph Chains, S.M. Goddard, K. Jeffay, The Computer Journal, Volume 44, Number 6, 2001, (special issue on high-assurance systems), pages 486-503.

Rate-Based Resource Allocation Methods for Embedded Systems, K. Jeffay, S.M. Goddard, in, *Embedded Software*, Proceedings of the First International Workshop on Embedded Software (*EMSOFT 2001*),

Tahoe City, CA, October 2001, Lecture Notes in Computer Science, Volume 2211, T. Henzinger, C. Kirsch, editors, Springer-Verlag, Berlin, Germany, 2001, pages 204-222.

Beyond Audio and Video: Multimedia Networking Support for Distributed, Immersive Virtual Environments, K. Jeffay, T. Hudson, M. Parris, Proceedings of the 27th EUROMICRO Conference, Warsaw, Poland, September 2001, pages 300-307.

Tuning RED for Web Traffic, M. Christiansen, K. Jeffay, D. Ott, F.D. Smith, IEEE/ACM Transactions on Networking, Volume 9, Number 3, (June 2001), pages 249-264.

What TCP/IP Protocol Headers Can Tell Us About the Web, F.D. Smith, F. Hernández Campos, K. Jeffay, D. Ott, Proceedings of ACM SIGMETRICS 2001/Performance 2001, Cambridge, MA, June 2001, pages 245-256.

Experiments in Best-Effort Multimedia Networking for a Distributed Virtual Environment, T. Hudson, M.C. Weigle, K. Jeffay, R.M. Taylor II, in *Multimedia Computing and Networking 2001*, Proceedings, SPIE Proceedings Series, Volume 4312, San Jose, CA, January 2001, pages 88-98.

Analyzing the Real-Time Properties of a U.S. Navy Signal Processing System, S.M. Goddard, K. Jeffay, International Journal of Reliability, Quality and Safety Engineering, Volume 8, Number 4, December 2001, (special issue of HASE '99 best papers) pages 301-322.

A Comparative Study of the Realization of Rate-Based Computing Services in General Purpose Operating Systems, K. Jeffay, G. Lamastra, Proceedings of the Seventh IEEE International Conference on Real-Time Computing Systems and Applications, Cheju Island, South Korea, December 2000, pages 81-90.

The Synthesis of Real–Time Systems from Processing Graphs, S.M. Goddard, K. Jeffay, Proceedings of the Fifth IEEE International Symposium on High Assurance Systems Engineering, Albuquerque, NM, November 2000, pages 177-186.

Tuning RED for Web Traffic, M. Christiansen, K. Jeffay, D. Ott, F.D. Smith, Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August-September 2000, pages 139-150.

Towards a Better-Than-Best-Effort Forwarding Service for Multimedia Flows, K. Jeffay, IEEE Multimedia, Volume 6, Number 4, October-December 1999, pages 84-88.

A Theory of Rate-Based Execution, K. Jeffay, S.M. Goddard, Proceedings of the 20th IEEE Real-Time Systems Symposium, Phoenix, AZ, December 1999, pages 304-314.

Parallel Switching in Connection-Oriented Networks, S. Baruah, K. Jeffay, J. Anderson, Proceedings of the 20th IEEE Real-Time Systems Symposium, Phoenix, AZ, December 1999, pages 200-209.

Analyzing the Real-Time Properties of a U.S. Navy Signal Processing System, S.M. Goddard, K. Jeffay, Proceedings of the Fourth IEEE International Symposium on High Assurance Systems Engineering, Washington, DC, November 1999, pages 141-150.

Application-Level Measurements of Performance on the vBNS, M. Clark, K. Jeffay, Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Volume 2, Florence, Italy, June 1999, pages 362-366.

Lightweight Active Router-Queue Management for Multimedia Networking, M. Parris, K. Jeffay, F.D. Smith, in *Multimedia Computing and Networking 1999*, Proceedings, SPIE Proceedings Series, Volume 3654, San Jose, CA, January 1999, pages 162-174.

Proportional Share Scheduling of Operating System Services for Real-Time Applications, K. Jeffay, F.D. Smith, A. Moorthy, J.H. Anderson, Proceedings of the 19th IEEE Real-Time Systems Symposium, Madrid, Spain, December 1998, pages 480-491.

Efficient Object Sharing in Quantum-Based Real-Time Systems, J.H. Anderson, R. Jain, K. Jeffay, Proceedings of the 19th IEEE Real-Time Systems Symposium, Madrid, Spain, December 1998, pages 346-355.

A Better-Than-Best-Effort Service for Continuous Media UDP Flows, M. Parris, K. Jeffay, F.D. Smith, J. Borgersen, Proceedings of the Eighth International Workshop on Network and Operating System Support for Digital Audio and Video, Cambridge, UK, July 1998, pages 193-197.

Managing Memory Requirements in the Synthesis of Real-Time Systems from Processing Graphs, S. Goddard, K. Jeffay, Proceedings of the Fourth IEEE Real-Time Technology and Applications Symposium, Denver, CO, June 1998, pages 59-70.

Fair On-Line Scheduling of a Dynamic Set of Tasks on a Single Resource, S.K. Baruah, J.E. Gehrke, C.G. Plaxton, I. Stoica, H. Abdel-Wahab, K. Jeffay, *Information Processing Letters*, Volume 64, Number 1, October 1997, pages 43-51.

A Two-Dimensional Audio Scaling Enhancement to an Internet Videoconferencing System, P. Nee, K. Jeffay, M. Clark, G. Danneels, Proceedings of the International Workshop on Audio-Visual Services over Packet Networks, Aberdeen, Scotland, UK, September 1997, pages 201-206.

Feasibility Concerns in PGM Graphs With Bounded Buffers, S. Baruah, S. Goddard, K. Jeffay, Proceedings of the Third IEEE International Conference on Engineering of Complex Computer Systems, Como, Italy, September 1997, pages 130-139.

Analyzing the Real-Time Properties of a Dataflow Execution Paradigm Using a Synthetic Aperture Radar Application, S. Goddard, K. Jeffay, Proceedings of the Third IEEE Real-Time Technology and Applications Symposium, Montreal, Canada, June 1997, pages 60-71.

Real-Time Computing with Lock-Free Shared Objects, J.H. Anderson, S. Ramamurthy, K. Jeffay, ACM Transactions on Computing Systems, Volume 15, Number 2, May 1997, pages 134-165.

The Performance of Two-Dimensional Media Scaling for Internet Videoconferencing, P. Nee, K. Jeffay, G. Danneels, Proceedings of the Seventh International Workshop on Network and Operating System Support for Digital Audio and Video, St. Louis, MO, May 1997, pages 237-248. Republished in "Readings in Multimedia Computing," K. Jeffay, H.J. Zhang, editors, Morgan Kaufmann, 2002.

On the Duality between Resource Reservation and Proportional Share Resource Allocation, I. Stoica, H. Abdel-Wahab, K. Jeffay, in *Multimedia Computing and Networking 1997*, Proceedings, SPIE Proceedings Series, Volume 3020, San Jose, CA, February 1997, pages 207-214.

Strategic Directions in Real-Time and Embedded Systems, J.A. Stankovic, A. Burns, K. Jeffay, M. Jones, G. Koob, I. Lee, J. Lehoczky, J. Liu, A. Mok, K. Ramamritham, J. Ready, L. Sha, A. van Tilborg, *ACM Computing Surveys*, Volume 28, Number 4, December 1996, pages 751-763.

A Proportional Share Resource Allocation Algorithm For Real-Time, Time-Shared Systems, I. Stoica, H. Abdel-Wahab, K. Jeffay, S.K. Baruah, J.E. Gehrke, C.G. Plaxton, Proceedings of the 17th IEEE Real-Time Systems Symposium, Washington, DC, December 1996, pages 288-299.

A General Framework For Continuous Media Transmission Control, T.M. Talley, K. Jeffay, Proceedings of the 21st IEEE Conference on Local Computer Networks, Minneapolis, MN, October 1996, pages 374-383.

A Router-Based Congestion Control Scheme For Real-Time Continuous Media, K. Jeffay, M. Parris, T. Talley, F.D. Smith, Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video, Zushi, Japan, April 1996, pages 79-86.

Lock-Free Transactions for Real-time Systems, J.H. Anderson, S. Ramamurthy, M. Moir, K. Jeffay, Proceedings of the First Workshop on Real-Time Databases: Issues and Applications, Newport Beach,

CA, March 1996, pages 107-114.

Real-Time Computing with Lock-Free Shared Objects, J.H. Anderson, S. Ramamurthy, K. Jeffay, Proceedings of the 16th IEEE Real-Time Systems Symposium, Pisa, Italy, December 1995, pages 28-37.

Early Experience with the Repository for Patterned Injury Data, D. Stotts, J.B. Smith, K. Jeffay, P. Dewan, D.K. Smith, W. Oliver, Proceedings of the SPIE International Symposium on Investigative and Trial Image Processing, San Diego, CA, July 1995, SPIE Volume 2567, 1995, pages 249-260.

Early Prototypes of the Repository for Patterned Injury Data, P. Dewan, K. Jeffay, J. Smith, D. Stotts, W. Oliver, Proceedings of Digital Libraries '95, The Second Annual Conference on the Theory and Practice of Digital Libraries, Austin, TX, June 1995, pages 123-130.

Support For Real-Time Computing Within General Purpose Operating Systems: Supporting co-resident operating systems, G. Bollella, K. Jeffay, Proceedings of the IEEE Real-Time Technology and Applications Symposium, Chicago, IL, May 1995, pages 4-14.

A Rate-Based Execution Abstraction For Multimedia Computing, K. Jeffay, D. Bennett, Proceedings of the Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, April 1995, published in *Lecture Notes in Computer Science*, T.D.C. Little, R. Gusella, editors, Volume 1018, pages 64-75, Springer-Verlag, Heidelberg, Germany, 1995.

An Empirical Study of Delay Jitter Management Policies, D.L. Stone, K. Jeffay, *ACM Multimedia Systems*, Volume 2, Number 6, January 1995, pages 267-279. Republished in "Readings in Multimedia Computing," K. Jeffay, H.J. Zhang, editors, Morgan Kaufmann, 2002.

On the Partitioning of Function in Distributed Synchronous Collaboration Systems, J. Menges, K. Jeffay, ACM CSCW '94, Proceedings of the Workshop on Distributed Systems, Multimedia, and Infrastructure Support in CSCW, Research Triangle Park, NC, October 1994, SIGOIS Bulletin, Volume 15, Number 2, December 1994, pages 34-37.

Two-Dimensional Scaling Techniques For Adaptive, Rate-Based Transmission Control of Live Audio and Video Streams, T.M. Talley, K. Jeffay, Proceedings of the Second ACM International Conference on Multimedia, San Francisco, CA, October 1994, pages 247-254.

Transport and Display Mechanisms For Multimedia Conferencing Across Packet-Switched Networks, K. Jeffay, D.L. Stone, F.D. Smith, *Computer Networks and ISDN Systems*, Volume 26, Number 10, July 1994, pages 1281-1304. Republished in "A Guided Tour of Multimedia Systems and Applications," B. Furht, M. Milenkovic, editors, IEEE Computer Society Press, 1995.

A Patterned Injury Digital Library for Collaborative Forensic Medicine, D. Stotts, J.B. Smith, P. Dewan, K. Jeffay, F.D. Smith, D. Smith, S. Weiss, J. Coggins, W. Oliver, Proceedings of Digital Libraries '94, The First Annual Conference on the Theory and Practice of Digital Libraries, College Station, TX, June 1994, pages 25-33.

The Artifact-Based Collaboration System: An infrastructure for supporting and studying collaboration, K. Jeffay, J.B. Smith, F.D. Smith, D.E. Shackelford, J. Menges, Proceedings of the 15th Interdisciplinary Workshop on Informatics and Psychology, Schärding, Austria, May 1994, 27 pages.

On Latency Management in Time-Shared Operating Systems, K. Jeffay, Proceedings of the 11th IEEE Workshop on Real-Time Operating Systems and Software, Seattle, WA, May 1994, pages 86-90.

Inverting X: An Architecture for a Shared Distributed Window System, J. Menges, K. Jeffay, Proceedings of the Third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Morgantown, WV, April 1994, IEEE Computer Society Press, pages 53-64.

Issues, Problems, and Solutions in Sharing X Clients on Multiple Displays, H. Abdel-Wahab, K. Jeffay, *Internetworking — Research and Practice*, Volume 5, Number 1, March 1994, pages 1-15.

Dynamic Participation in a Computer-based Conferencing System, G. Chung, K. Jeffay, H. Abdel-Wahab, *Computer Communications*, Volume 17, Number 1, January 1994, pages 7-16.

Accounting for Interrupt Handling Costs in Dynamic Priority Task Systems, K. Jeffay, D.L. Stone, Proceedings of the 14th IEEE Real-Time Systems Symposium, Raleigh-Durham, NC, December 1993, pages 212-221.

Queue Monitoring: A Delay Jitter Management Policy, D.L. Stone, K. Jeffay, Proceedings of the Fourth International Workshop on Network and Operating System Support for Digital Audio and Video, Lancaster, UK, November 1993, published in *Lecture Notes in Computer Science*, D. Shepherd, G. Blair, G. Coulson, N. Davies, F. Garcia, editors, Volume 846, pages 149-160, Springer-Verlag, Heidelberg, Germany, 1994.

The Real-Time Producer/Consumer Paradigm: A paradigm for the construction of efficient, predictable real-time systems, K. Jeffay, Proceedings of the 1993 ACM/SIGAPP Symposium on Applied Computing, Indianapolis, IN, ACM Press, February 1993, pages 796-804.

Accommodating Late-Comers in Shared Window Systems, G. Chung, K. Jeffay, H. Abdel-Wahab, *IEEE Computer*, Volume 26, Number 1, January 1993, pages 72-74.

Scheduling Sporadic Tasks with Shared Resources in Hard-Real-Time Systems, K. Jeffay, Proceedings of the 13th IEEE Real-Time Systems Symposium, Phoenix, AZ, December 1992, pages 89-99.

Adaptive, Best-Effort, Delivery of Audio and Video Data Across Packet-Switched Networks, K. Jeffay, D.L. Stone, T. Talley, F.D. Smith, Proceedings of the Third International Workshop on Network and Operating System Support for Digital Audio and Video, La Jolla, CA, November 1992, published in *Lecture Notes in Computer Science*, V. Rangan, editor, Volume 712, pages 3-14, Springer-Verlag, Heidelberg, Germany, 1993.

Architecture of the Artifact-Based Collaboration System Matrix, K. Jeffay, J.K. Lin, J. Menges, F.D. Smith, J.B. Smith, ACM CSCW '92, Proceedings of the Conference on Computer-Supported Cooperative Work, Toronto, Canada, ACM Press, November 1992, pages 195-202.

Client/Server Protocol Filtering: An infrastructure for supporting group collaborations, K. Jeffay, J. Menges, J.-K. Lin, ACM CSCW '92, Proceedings of the Workshop on Tools and Technologies, Toronto, Canada, November 1992, pages 96-99.

Kernel Support for Live Digital Audio and Video, K. Jeffay, D.L. Stone, F.D. Smith, *Computer Communications*, Volume 15, Number 6, July/August 1992, pages 388-395.

On Kernel Support for Real-Time Multimedia Applications, K. Jeffay, Proceedings of the Third IEEE Workshop on Workstation Operating Systems, Key Biscayne, FL, April 1992, pages 39-46.

On Non-Preemptive Scheduling of Periodic and Sporadic Tasks, K. Jeffay, D.F. Stanat, C.U. Martel, Proceedings of the Twelfth IEEE Real-Time Systems Symposium, San Antonio, TX, December 1991, pages 129-139.

Kernel Support for Live Digital Audio and Video, K. Jeffay, D.L. Stone, F.D. Smith, Proceedings of the Second International Workshop on Network and Operating System Support for Digital Audio and Video, Heidelberg, Germany, November 1991, published in *Lecture Notes in Computer Science*, R.G. Herrtwich, editor, Volume 614, pages 10-21, Springer-Verlag, Heidelberg, Germany, 1992.

UNC Collaboratory Project Overview, J.B. Smith, F.D. Smith, P. Calingaert, J.R. Hayes, D. Holland, K. Jeffay, M. Lansman, Proceedings of the 1991 Symposium on Command and Control Research, National Defense University, Washington, D.C., June 1991, pages 341-391.

YARTOS: Kernel support for efficient, predictable real-time systems, K. Jeffay, D. Stone, D. Poirier, Proceedings of the Joint Eighth IEEE Workshop on Real-Time Operating Systems and Software and IFAC/IFIP Workshop on Real-Time Programming, Atlanta, GA, May 1991, in Real-Time Systems

Newsletter, Volume 7, Number 4, Fall 1991, pages 8-13. Republished in "Real-Time Programming," W. Halang, K. Ramamritham, editors, 1992.

System Design for Workstation-Based Conferencing With Digital Audio and Video, K. Jeffay, F.D. Smith, Proceedings of the IEEE Conference on Communication Software: Communications for Distributed Applications and Systems, Chapel Hill, NC, April 1991, pages 169-180.

Designing a Workstation-Based Conferencing System Using the Real-Time Producer/Consumer Paradigm, K. Jeffay, F.D. Smith, Proceedings of the First International Workshop on Network and Operating System Support for Digital Audio and Video, International Computer Science Institute, Berkeley, CA, November 1990, pages 40-55.

Analysis of a Synchronization and Scheduling Discipline for Real-Time Tasks with Preemption Constraints, K. Jeffay, Proceedings of the Tenth IEEE Real-Time Systems Symposium, Santa Monica, CA, December 1989, pages 295-305.

Corset & Lace: Adapting Ada Runtime Support to Real-Time Systems, T.P. Baker, K. Jeffay, Proceedings of the Eighth IEEE Real-Time Systems Symposium, San Jose, CA, December 1987, pages 158-167.

Research in Real-Time Systems, A.C. Shaw, C. Binding, W.L. Hu, K. Jeffay, Proceedings of the Third IEEE Workshop on Real-Time Operating Systems, Boston, MA, February 1986, pages 121-131.

Book Chapters

Rate-Based Resource Allocation Methods, K. Jeffay, in, "Handbook of Real-Time and Embedded Systems," I. Lee, J. Y-T. Leung, S.H. Son, editors, Chapman & Hall/CRC Press, Boca Raton, FL, 2008, pages 4-1 – 4-15.

Visualization and Natural Control Systems for Microscopy, R.M. Taylor II, D. Borland, F.P. Brooks Jr., M. Falvo, M. Guthold, T. Hudson, K. Jeffay, G. Jones, D. Marshburn, S.J. Papadakis, L.-C. Qin, A. Seeger, F.D. Smith, D.H. Sonnenwald, R. Superfine, S. Washburn, C. Weigle, M.C. Whitton, P. Williams, L. Vicci, W. Robinett, in "Visualization Handbook," C. Johnson, C. Hansen, editors, Harcourt Academic Press, 2005, pages 875-900.

An Empirical Study of Delay Jitter Management Policies, D.L. Stone, K. Jeffay, in "Readings in Multimedia Computing and Networking," K. Jeffay, H.J. Zhang, editors, Morgan Kaufmann, San Francisco, CA, 2002, pages 525-537.

The Performance of Two-Dimensional Media Scaling for Internet Videoconferencing, P. Nee, K. Jeffay, G. Danneels, in "Readings in Multimedia Computing and Networking," K. Jeffay, H.J. Zhang, editors, Morgan Kaufmann, San Francisco, CA, 2002, pages 581-592.

Lock-Free Transactions for Real-time Systems, J.H. Anderson, S. Ramamurthy, M. Moir, K. Jeffay, in "Real-Time Database Systems: Issues and Applications," A. Bestavros, K.J. Lin, S.H. Son, editors, Kluwer Academic Publishers, Norwell, MA, 1997, pages 215-234.

Transport and Display Mechanisms For Multimedia Conferencing Across Packet-Switched Networks, K. Jeffay, D.L. Stone, F.D. Smith, in "A Guided Tour of Multimedia Systems and Applications," B. Furht, M. Milenkovic, editors, IEEE Computer Society Press, Los Alamitos, CA, 1995, pages 439-461.

Contributor to: *R&D for the NII: Technical Challenges*, M.K. Vernon, E.D. Lazowska, S.D. Personick, editors, Interuniveristy Communications Council (EDUCOM), 1994.

YARTOS: Kernel support for efficient, predictable real-time systems, K. Jeffay, D. Stone, D. Poirier, in "Real-Time Programming," W. Halang, K. Ramamritham, editors, Pergamon Press, Oxford, UK, 1992, pages 7-12.

Videotapes

Adaptive, Best-Effort Delivery of Live Audio and Video Across Packet-Switched Networks, K. Jeffay, D.L. Stone, ACM Multimedia '94 Video Proceedings, San Francisco, CA, October 1994, 6 minutes. Excerpts also appear on the CD-ROM version of the conference proceedings.

Abstracts & Short Papers

Quantifying the Effects of Recent Protocol Improvements to Standards-Track TCP, M.C. Weigle, K. Jeffay, and F.D. Smith, Proceedings of the 11th IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS), Orlando, FL, October 2003, pages 226-229.

Engineering of Rate Based Services for Real-Time Computing, G. Lamastra, K. Jeffay, 20th IEEE Real-Time Systems Symposium Work in Progress Proceedings, Phoenix, AZ, December 1999, pages 51-55.

Lightweight Active Router-Queue Management for Multimedia Networking, M. Parris, K. Jeffay, F.D. Smith, IS&T/SPIE International Symposium on Electronic Imaging: Science and Technology, San Jose, CA, January 1999, pages 280-281.

Support For Real-Time Computing in Windows NT, K. Jeffay, Usage Abstracts, USENIX Windows NT Workshop, Seattle, WA, August 1997, page 84.

On the Duality between Resource Reservation and Proportional Share Resource Allocation, I. Stoica, H. Abdel-Wahab, K. Jeffay, IS&T/SPIE Ninth Symposium on Electronic Imaging: Science and Technology 1997, San Jose, CA, February 1997, pages 135-136.

Technical and Educational Challenges For Real-Time Computing, K. Jeffay, *ACM Computing Surveys*, Volume 28A, Number 4(es), December 1996, URL *http://www.acm.org/pubs/contents/journals/surveys/1996-28/#4es*, 3 pages.

Distributed Real-Time Dataflow: An Execution Paradigm for Image Processing and Anti-Submarine Warfare Applications, S. Goddard, K. Jeffay, 19th IEEE Real-Time Systems Symposium Work in Progress Proceedings, Washington, DC, December 1996, pages 55-58.

Why Using the Request Abstraction in Proportional Share Allocation Systems is Useful, I. Stoica, H. Zhang, K. Jeffay, Proceedings of the IEEE Real-Time Systems Symposium Workshop on Resource Allocation Problems in Multimedia Systems, Washington, DC, December 1996, 4 pages.

Future distributed embedded and real-time applications will be adaptive: Meanings, challenges, and research paradigms, A.K. Mok, C.L. Heitmeyer, K. Jeffay, M.B. Jones, C.D. Locke, R. Rajkumar, 15th Proceedings of the 15th IEEE International Conference on Distributed Computing Systems, Vancouver, British Columbia, Canada, May 1995, pages 182-184.

Adaptive, Best-Effort Delivery of Live Audio and Video Across Packet-Switched Networks, K. Jeffay, D.L. Stone, Proceedings of the Second ACM International Conference on Multimedia, San Francisco, CA, October 1994, pages 487-488.

Adaptive Rate-Based Flow and Latency Management of Audio and Video Streams, T.-M. Chen, T. Talley, K. Jeffay, Abstracts of the Fourth Workshop on Network and Operating System Support for Digital Audio and Video, Lancaster, UK, November 1993, pages 17-20.

Client/Server Protocol Filtering: An infrastructure for supporting group collaborations, K. Jeffay, J. Menges, J.-K. Lin, ACM CSCW '92, Proceedings of the Workshop on Tools and Technologies, Toronto, Canada, November 1992, pages 96-99.

Network and Operating Systems Support for Digital Audio and Video, K. Jeffay, in 13th ACM Symposium on Operating System Principles "Work in Progress" Abstracts, E.D. Lazowska, editor, Operating Systems Review, Volume 26, Number 2, April 1992, pages 7-31.

Posters

Efficient Management of a High-Capacity Airborne Network of Commercial Aircraft, B. Newton, J. Aikat, K. Jeffay, NSF Cyber-physical Systems Young Professional Workshop, Washington, DC, March 2014.

Variability of TCP Round-trip Times (RTTs) Over the Years, W. Yau, J. Aikat, K. Jeffay, Grace Hopper Celebration of Women in Computing (GHC), Baltimore, MD, October 2012.

Characterizing Modern Web Traffic using TCP Headers, S. Zolayvar, J. Aikat, K. Jeffay, Grace Hopper Celebration of Women in Computing (GHC), Baltimore, MD, October 2012.

Multivariate SVD Analyses For Network Anomaly Detection, J. Terrell, L. Zhang, K. Jeffay, A. Nobel, H. Shen, F.D. Smith, Z, Zhu, ACM SIGCOMM 2005 Poster Session, Philadelphia, PA, August 2005.

How Real Can Synthetic Traffic Be?, F. Hernández-Campos, K. Jeffay, F.D. Smith, ACM SIGCOMM 2004 Poster Session, Portland, OR, August 2004.

A Non-Parametric Approach to Generation and Validation of Synthetic Network Traffic, F. Hernández-Campos, A. Nobel, F.D. Smith, K. Jeffay, IMA Workshop on Measurement, Modeling, and Analysis of the Internet, Minneapolis, MN, January 2004.

Sync-TCP: Using GPS Synchronized Clocks for Early Congestion Detection in TCP, M.C. Weigle, K. Jeffay, F.D. Smith, ACM SIGCOMM 2000 Poster Session, Stockholm, Sweden, August 2000, page 6.

Reviews

Advance Reservation Systems, K. Jeffay, Proceedings of the Fifth International Workshop on Network and Operating System Support for Digital Audio and Video, Durham, NH, April 1995, published in *Lecture Notes in Computer Science*, T.D.C. Little, R. Gusella, editors, Volume 1018, pages 1-2, Springer-Verlag, Heidelberg, Germany, 1995.

Books and Proceedings

The Effects of Traffic Structure on Application and Network Performance, J. Aikat, K. Jeffay, and F. D. Smith, Springer Publishing, New York, NY, 2012, 289 pages.

The Effects of Active Queue Management on TCP Application Performance, An experimental performance evaluation, L. Le, K. Jeffay, F.D. Smith, Lambert Academic Publishing, Berlin, Saarbrücken, Germany, 2009, 222 pages.

Quality-of-Service — IWQoS 2003, K. Jeffay, I. Stoica, K. Wehrle, editors, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, Germany, Volume 2707, 2003, ISBN 3-540-40281-0, 517 pages.

Readings in Multimedia Computing and Networking, K. Jeffay, H.J. Zhang, editors, Morgan Kaufman, San Francisco, CA, 2002, ISBN 1-55860-651-3, 863 pages.

Proceedings, The 22nd IEEE Real-Time Systems Symposium, K. Jeffay, G. Buttazzo, editors, IEEE Computer Society Press, Los Alamitos, CA, 2001, ISBN 0-7695-1420-0, 321 pages.

Proceedings, The Sixth IEEE International Symposium on Computers and Communications, K. Jeffay, R. Steinmetz, editors, IEEE Computer Society Press, Los Alamitos, CA, 2001, ISBN 0-7695-1177-5, 754 pages.

Proceedings, The 21st IEEE Real-Time Systems Symposium, K. Jeffay, W. Zhao, editors, IEEE Computer Society Press, Los Alamitos, CA, 2000, ISBN 0-7695-0900-2, 311 pages.

Proceedings, The 10th International Workshop on Network and Operating System Support for Digital Audio and Video, K. Jeffay and H. Vin, editors, Technical Report, Department of Computer Science, University of North Carolina at Chapel Hill, 2000, 320 pages. Proceedings also published on-line at *http://www.cs.unc.edu/nossdav2000*.

ACM Multimedia '99, D. Bulterman, K. Jeffay, H.J. Zhang, editors, ACM Press, Los Angeles, CA, 1999, ISBN 0-8194-3125-7, 500 pages.

Multimedia Computing and Networking 1999, D.D. Kandlur, K. Jeffay, T. Roscoe, editors, Proceedings of SPIE, Volume 3654, SPIE, Bellingham, WA, 1998, ISBN 0-8194-3125-7, 328 pages.

Multimedia Computing and Networking 1998, K. Jeffay, D.D. Kandlur, T. Roscoe, editors, Proceedings

of SPIE, Volume 3310, SPIE, Bellingham, WA, 1997, ISBN 0-8194-250-0, 266 pages.

Proceedings, 1997 IEEE Real-Time Technology and Applications Symposium, R. Rajkumar, K. Jeffay, editors, IEEE Computer Society Press, Los Alamitos, CA, 1997, ISBN 0-8186-8016-4, 269 pages.

Proceedings, IEEE Real-Time Systems Symposium Workshop on Resource Allocation Problems in Multimedia Systems, K. Jeffay, editor, *http://www.cs.unc.edu/~jeffay/meetings/mm-wrkshp96/prog. html,* 1996, 250 pages.

Proceedings, 1996 IEEE Real-Time Technology and Applications Symposium, K. Jeffay, W. Zhao, editors, IEEE Computer Society Press, Los Alamitos, CA, 1996, ISBN 0-8186-7448-2, 264 pages.

Unrefereed Publications

Invited Papers

Modeling and Generation of TCP Application Workloads, F. Hernández-Campos, K. Jeffay, F.D. Smith, Proceedings of the 4th IEEE International Conference on Broadband Communications, Networks, and Systems, Raleigh, NC, September 2007, 10 pages.

Statistical Clustering of Internet Communication Patterns, F. Hernández-Campos, A.B. Nobel, F.D. Smith, K. Jeffay, Proceedings of the 35th Symposium on the Interface of Computing Science and Statistics, Salt Lake City, UT, July 2003, Computing Science and Statistics, Volume 35, 2004.

Rate-Based Resource Allocation Methods for Embedded Systems, K. Jeffay, S.M. Goddard, in, *Embedded Software*, Proceedings of the First International Workshop on Embedded Software (*EMSOFT 2001*), Tahoe City, CA, October 2001, Lecture Notes in Computer Science, Volume 2211, T. Henzinger, C. Kirsch, editors, Springer Verlag, Berlin, Germany, 2001, pages 204-222.

Beyond Audio and Video: Multimedia Networking Support for Distributed, Immersive Virtual Environments, K. Jeffay, T. Hudson, M. Parris, Proceedings of the 27th EUROMICRO Conference, Workshop on Multimedia and Telecommunication, Warsaw, Poland, September 2001, pages 300-307.

Experiments in Best-Effort Multimedia Networking for a Distributed Virtual Environment, T. Hudson, M.C. Weigle, K. Jeffay, R.M. Taylor II, in *Multimedia Computing and Networking 2001*, Proceedings, SPIE Proceedings Series, Volume 4312, San Jose, CA, January 2001, pages 88-98.

Towards a Better-Than-Best-Effort Forwarding Service for Multimedia Flows, K. Jeffay, IEEE Multimedia, Volume 6, Number 4, October-December 1999, pages 84-88.

Network Support For Distributed, Immersive, Virtual Laboratories K. Jeffay, Proceedings of the NSF Workshop on Automated Control of Distributed Instrumentation, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign, Urbana, IL, April 1999, pages 73-78.

Storage Requirements For Distributed Virtual Laboratories, K. Jeffay, Proceedings of the Internet2 Distributed Storage Infrastructure Application Workshop, University of North Carolina at Chapel Hill, Chapel Hill, NC, March 1999, pages 57-59.

Efficient Kernel Support for Continuous Time Media Systems, K. Jeffay, Proceedings of the ITC Workshop on Continuous Time Media, Carnegie Mellon University, Pittsburgh, PA, June 1991, 4 pages.

White Papers

Contributor to: *NSF Report on Network Research Testbeds*, B. Braden, M. Gerla, J. Kurose, J. Lepreau, R. Rao, J. Turner, editors, December 2002. A white paper commission by NSF that led to the creation of a $10 million funding program for network research testbeds.

Contributor to: *R&D for the National Information Infrastructure: Technical Challenges*, M.K. Vernon, E.D. Lazowska, S.D. Personick, editors, Interuniveristy Communications Council (EDUCOM), 1994.

Technical

Beyond Window Sharing Hacks: Support for First-Class Window Sharing, J. Menges, K. Jeffay, Report

Reports TR97-021, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, April 1997.

A Proportional Share Resource Allocation Algorithm For Real-Time, Time-Shared Systems, I. Stoica, H. Abdel-Wahab, K. Jeffay, S.K. Baruah, J.E. Gehrke, C.G. Plaxton, Report TR96-038, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, September 1996.

Predicting Worst Case Execution Times on a Pipelined RISC Processor, S.J. Bharrat, K. Jeffay, Report TR94-072, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, April 1994.

The Design, Implementation, and Use of a Sporadic Tasking Model, K. Jeffay, D. Becker, D. Bennett, S. Bharrat, T. Gramling, M. Housel, Report TR94-073, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, April, 1994.

UNC Collaboratory Project Overview, J.B. Smith, F.D. Smith, P. Calingaert, J.R. Hayes, D. Holland, K. Jeffay, M. Lansman, Report 90-042, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, November 1990.

An Implementation and Application of the Real-Time Producer/Consumer Paradigm, D. Poirier, K. Jeffay, Report 90-038, University of North Carolina at Chapel Hill, Department of Computer Science, Chapel Hill, NC, October 1990.

The Real-Time Producer/Consumer Paradigm: Towards Verifiable Real-Time Computations, K. Jeffay, Report 89-09-15, University of Washington, Department of Computer Science, Seattle, WA, September 1989, (Ph.D. Dissertation).

On Optimal, Non-Preemptive Scheduling of Periodic and Sporadic Tasks, K. Jeffay, R. Anderson, Report 88-11-06, University of Washington, Department of Computer Science, Seattle, WA, November 1988.

On Optimal, Non-Preemptive Scheduling of Periodic Tasks, K. Jeffay, Report 88-10-03, University of Washington Department of Computer Science, Seattle, WA, October 1988.

Software Engineering of Real-Time Operating Systems, A.C. Shaw, K. Jeffay, Report 88-01-01, University of Washington Department of Computer Science, Seattle, WA, January 1988.

Concurrent Programming with Time, A thesis proposal, K. Jeffay, Report 87-10-03, University of Washington Department of Computer Science, Seattle, WA, October 1987.

A Lace for Ada's Corset, T.P. Baker, K. Jeffay, Report 86-09-06, University of Washington Department of Computer Science, Seattle, WA, September 1986.

Software Distributions

GENI tmix — A synthetic TCP workload generator for GENI, J. Aikat, D. O'Neil, B. Newton, K. Jeffay, June 2012.

Distributed to the GENI networking research community.

tmix — A synthetic TCP workload generator, F. Hernández Campos, K. Jeffay, F.D. Smith, June 2005.

Distributed to other academic and industry networking research groups via the Web.

UNC VNC — A version of the public domain workspace sharing system VNC to support secure sharing of workspace regions and windows of individual applications. J. Branscomb, L. Fowler, K. Jeffay, August 2004.

Distributed to other research groups and the public via the Web and SourceForge.

thttp 2003 — An HTTP/v1.0 and v1.1 traffic generation program used to simulate the traffic generated by a collection of geographically distributed web browsers and servers. L. Le, A. Van Osdol, F.D. Smith, K. Jeffay, January 2004.

Distributed to other academic and industry networking research groups via the Web.

thttp 2001 — An HTTP/v1.0 and v1.1 traffic generation program used to simulate the traffic generated by a collection of geographically distributed web browsers and servers. F. Hernández Campos, F.D. Smith, K. Jeffay, August 2002.

Distributed to other academic and industry networking research groups via the Web.

Packmime Implementation in ns — An implementation of the Lucent Bell Labs (Bill Cleaveland) HTTP traffic model in ns. M.C. Weigle, K. Jeffay, F.D. Smith, September 2001.

Distributed as part of the Network Simulator (**ns-2**) distribution.

UNC Campus Network Trace Data and Empirical Distributions — A collection of network traces and empirical distributions illustrating the nature of web browsing and the structure of web pages. F. Hernández Campos, K. Jeffay, F.D. Smith, June 2001.

Distributed to other academic and industry networking research groups via the Web.

HTTP Analysis Tools— A set of tools for analyzing http connections. F. Hernández Campos, K. Jeffay, F.D. Smith, June 2001.

Distributed as part of the Network Simulator (**ns-2**) distribution.

thttp — An HTTP traffic generation program used to simulate the traffic generated by a collection of geographically distributed web browsers and servers. M. Christiansen, F.D. Smith, K. Jeffay, January 2000.

Distributed to other academic and industry networking research groups via the Web.

FlowGen — A general purpose, programmable network flow generator. M. Clark, K. Jeffay, F.D. Smith, December 1999.

Distributed to Advanced Networking Systems Inc. for use in performance evaluation studies of Internet 2 and Abilene. Also available via anonymous *ftp* from *http://www.cs.unc.edu/Research/Dirt*.

TruTime Device Drivers for FreeBSD — A set of FreeBSD device drivers for the TruTime GPS boards, A. Moorthy, K. Jeffay, and F.D. Smith, December 1998.

Distributed to Advanced Networking Systems Inc. for use in the IETF Internet Performance Measurement Initiative (IPMI) Surveyor network.

YARTOS (*Yet Another Real-Time Operating System*) — A real-time operating system kernel for Intel x86 platforms that uses a novel task implementation based on a single, shared run-time stack, an earliest-deadline-first processor scheduling algorithm developed at UNC, and integrates processor scheduling and inter-process communication. K. Jeffay and D. Stone, January 1996.

Distributed to several universities for use in operating system research and teaching. Also adopted by researchers at IBM's Networking Software Division, RTP, NC, for use in constructing a network protocol evaluation testbed.

XPEL (*The X Protocol Engine Library*) — A C^{++} library for constructing modular X Window System pseudo-servers, J. Menges and K. Jeffay, November 1993.

Distributed on the Internet via anonymous *ftp* from *ftp.cs.unc.edu*.

XTV (*X Terminal View*) — An X Window System pseudo-server that supports window sharing and conferencing across the Internet, H. Abdel-Wahab (Old Dominion University) and K. Jeffay, August 1991.

Distributed by MIT as part of the contributed software portion of the X Window System distribution (release 11, version 5). Last revised January 1993.

Also adopted by the NEC Corporation of Japan as the basis for their *Xprotodist* (X protocol distributor) product.

Patents

High-Performance Virtualized Applications Platform, U.S. Patent Application No. 421-343. A. Blate, K. Jeffay, May 2014.

Methods, Systems, and Computer Program Products For Network Server Performance Anomaly Detection, U.S. Patent No. 8,938,532, J.S. Terrell. K. Jeffay, F.D. Smith, R. Broadhurst, January 2015.

Method for Understanding the Use of TCP/IP Networks by Users, and Non-Parametric Generation of Synthetic Internet Traffic, U.S. Patent Number 7,447,209, F. Hernández-Campos, K. Jeffay, F.D. Smith, A. Nobel, November 2008.

User Controlled Adaptive Flow Control for Packet Networks, U.S. Patent Number 5,892,754, V. Kompella, F.D. Smith, J.P. Gray, and K. Jeffay, April 1999.

Public Demonstrations

Education Modules for Networking, Cloud Computing and Security in Systems Courses, J. Aikat, K. Jeffay, Demo session at the ACM SIGCSE, Special Interest Group for Computer Science Education, Memphis, TN, March 2016.

Exploring the Internet's Future with GENI (Global Environment for Network Innovations), J. Aikat, K. Jeffay, NSF Higher Education Technology Conference 2013, Morgantown, WV, October 2013.

Your first experiment on GENI: a Hands-on Tutorial, J. Aikat, K. Jeffay, NSF Higher Education Technology Conference 2013, Morgantown, WV, October 2013.

NSF Workshop on GENI in Education (one day), J. Aikat, K. Jeffay, New York University - Polytechnic, NY, October 2013.

Using the GENI Networking Testbed, J. Aikat, K. Jeffay, ACM SIGCSE, Special Interest Group for Computer Science Education, Denver, Colorado, March 2013.

Run your Systems/Networking Experiments using (free) GENI Resources, Lunchtime Table Topics, S. Zolayvar, W. Yau, J. Aikat, K. Jeffay, Grace Hopper Celebration of Women in Computing, Baltimore, Maryland, October 2012.

NSF Workshop on Designing Tools and Curricula for Undergraduate Courses in Distributed Systems, J. Aikat, K. Jeffay, Boston, MA, July 2012.

Introduction to Network Experiments using the GENI CyberInfrastructure, J. Aikat, K. Jeffay, ACM SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems, London, UK, June 2012.

The nanoManipulator "reverse field trip" to Orange County High-School, G. Jones, R. Superfine, R.M. Taylor, M. Whitton, T. Lovelace, R. Parameswaran, K. Jeffay, F.D. Smith, Orange County High School, Hillsboro, NC, November 1999.

The Distributed NanoManipulator Project, R.M. Taylor, K. Jeffay, T. Hudson, M. Clark, an invited demonstration for the Internet 2 Spring 1999 Members Meeting, Washington, DC, April 1999.

A Demonstration of Internet Access to UNC-CH Advanced Microscopy Facilities for Science Education Outreach, K. Jeffay, F.D. Smith, R. Superfine, G. Jones, R.M. Taylor, T. Hudson, M. Clark, Orange County High School, Hillsboro, NC, June 1998.

The Artifact-Based Collaboration (ABC) System, J.B. Smith, K. Jeffay, D. Shackelford, J. Menges, J. Hilgedic, B. Ladd, M. Parris, E. Kupstas, T. Ellis, and T. Hudson, CSCW '94, ACM Conference on Computer-Supported Cooperative Work, RTP, NC, October 1994.

CONCUR: A Window System Supporting Window Sharing, J. Menges and K. Jeffay, CSCW '94, ACM

Conference on Computer-Supported Cooperative Work, RTP, NC, October 1994.

Adaptive, Best-Effort Delivery of Live Audio and Video Across Packet-Switched Networks, K. Jeffay, Second MCNC/NSF Packet Video Workshop, Research Triangle Park, NC, December 1992.

Grants and Awards

Active *GENI in the classroom: Course Modules for Teaching Networking Concepts*, J. Aikat, K. Jeffay, National Science Foundation (GENI Project: Raytheon Corporation), October 2013, 3 years, $178,978.

II-New: Seeing the Future: Ubiquitous Computing in EyeGlasses, J.-M. Frahm, *et al*., National Science Foundation, November 2013, 3 years, $613,623.

Completed *GENI Tutorial at ACM SIGCSE*, J. Aikat, K, Jeffay, National Science Foundation, grant number CNS 13-32340, March 2013, 1 year, $13,388.

Development of Education and Training Resources for GENI Experimenters, K, Jeffay, J. Aikat, Global Environment for Network Innovations/National Science Foundation, December 2011, 3 years, $388,590.

Exploring Privacy Breaches in Encrypted VoIP Communications, F. Monrose, K, Jeffay, National Science Foundation, grant number CNS 10-17318, August 2010, 3 years, $496,492.

Synthetic Traffic Generation Tools and Resource: A Community Resource for Experimental Networking Research, K, Jeffay, F.D. Smith, UNC, M. Weigle, Old Dominion University, A. Vahdat, University of California San Diego, P. Barford, University of Wisconsin, National Science Foundation, grant number CRI 07-09081, August 2007, 3 years, $799,745.

Modeling and Testing of Application Workloads on Corporate Enterprise Networks, K, Jeffay, F.D. Smith, IBM Global Services Faculty Award, September 2006, 2 years, $70,000 (exempt from indirect costs).

Tera-pixels: Using High-resolution Pervasive Displays to Transform Collaboration and Teaching, K, Jeffay, A. Lastra, K. Mayer-Patel, L. McMillan, F.D. Smith, National Science Foundation (CISE RI Program), grant number EIA 03-03590, August 2003, 5 years, $962,902.

Extracting and Using Semantic Information in Network Workloads, K, Jeffay, F.D. Smith, IBM Global Services Faculty Award, September 2006, 1 year, $40,000 (exempt from indirect costs).

Modeling and Testing of Application Workloads on Corporate Enterprise Networks, K, Jeffay, F.D. Smith, IBM Global Services Faculty Award, September 2006, 1 year, $40,000 (exempt from indirect costs).

Generation and Validation of Synthetic Internet Traffic, K, Jeffay, F.D. Smith, A.B. Noble, National Science Foundation, grant number ANI 03-23648, September 2003, 3 years, $470,000.

Rate-Based Resource Allocation Methods for Real-Time Embedded Systems, K, Jeffay, F.D. Smith, National Science Foundation, grant number CCR 02-08924, August 2002, 3 years, $179,990.

Empirical Workload Characterizations for Advanced Networks, K. Jeffay, F.D. Smith, Cisco Systems, November 2000, 3 years, $212,500.

Support for Active Queue Management, Distributed XP Programming, and a Teaching Laboratory for Web Services, F.D. Smith, K. Jeffay, J.B. Smith, P.D. Stotts, IBM Shared University Research Grant, August 2002, 1 year, $83,000.

Internet Traffic Measurement and Analysis and A Teaching Laboratory for Enterprise Java Computing, F.D. Smith, K. Jeffay, J.B. Smith, P.D. Stotts, IBM Shared University Research Grant, April 2001, 1

year, $83,000.

An NS Implementation of the HTTP Connection Model, F.D. Smith, K. Jeffay, M. Clark Weigle, Lucent Technologies, April 2001, $10,000.

Multimedia Networking Research in Support of Virtual Field Trips, K. Jeffay, F.D. Smith, R.M. Taylor II, Dell Computer Corporation Strategic Technology and Research (STAR) program, December 2000, 1 year, $25,000.

Rate-based Scheduling Technology for Latency-Sensitive Graphics Applications, J. Anderson, S. Baruah, K. Jeffay, R. Taylor, National Science Foundation, grant number ITR 00-82866, September 2000, 3.5 years, $350,000.

Active Router Queue Management for Congestion Control and Quality-of-Service, K. Jeffay, F.D. Smith, National Science Foundation, grant number ITR 00-82870, September 2000, 3 years, $451,903.

Internet Measurements and Analysis to Support the nanoManipulator and Tele-Immersion, F.D. Smith, K. Jeffay, P. Jones, IBM Shared University Research Grant, September 2000, 1 year, $250,000.

The Performance of Differentiated Services Implementations for Supporting Distributed Virtual Environment Applications, K. Jeffay, North Carolina Network Initiative, September 2000, 1 year, $25,000.

Empirical Application-Workload Characterizations for Advanced Networks, K. Jeffay, F.D. Smith, Sun Microsystems, September 2000, $79,000.

Empirical Application-Workload Characterizations for Advanced Networks, F.D. Smith, K. Jeffay, MCNC, September 2000, 1 year, $50,000.

A Quality-of-Service Network for the nanoManipulator, K. Jeffay, Cabletron Inc. and the North Carolina Network Initiative, September 1999, $25,000.

Interactive Graphics for Molecular Studies and Microscopy — Supplement for Collaborative Science, F.P. Brooks, D. Erie (Chemistry), K. Jeffay, J. Samulski (Gene Therapy), F.D. Smith, D. Sonnenwald (Information and Library Science), R. Superfine (Physics and Astronomy), R. Taylor, National Institute of Health, October 1998, 4 years, $1,902,544.

Computing Power for Collaborative Science, S.R. Aylward, G. Bishop, D.W. Brenner, E. Bullitt, E.L. Chaney, V.L. Chi, B.J. Dempsey, N. England, A.G. Gash, D. Fritsch, H. Fuchs, B. Hemminger, J. Hermans, K. Jeffay, K. Keller, A. Lastra, M. Lin, D. Manocha, L.S. Nyland, S.M. Pizer, J.W. Poulton, J.F. Prins, J. Rosenman, F.D. Smith, R. Superfine, R.M. Taylor, S. Tell, G. Tracton, A. Tropsha, S. Washburn, G. Welch, Intel Corporation, August 1998, 3 years, $2,858,747. PI on *Multimedia Networking* and *A Distributed Teaching Laboratory for Networking and Internet Technologies* sections (F.D. Smith, B.J. Dempsey, Co-PIs), $557,669.

Empirical Application-Workload Characterizations for Advanced Networks, F.D. Smith, K. Jeffay, North Carolina Network Initiative, September 1998, $40,000.

Congestion control for high-speed networks, F.D. Smith, K. Jeffay, IBM Corporation, 1998, $20,000.

A Communications Middleware For Immersive Distributed Virtual Environments, K. Jeffay, North Carolina Network Initiative, 1998, $15,000.

Technology for Real-Time Services in Protocol Stack Implementations, K. Jeffay, IBM Corporation, 1997, $80,000.

Internet Access to UNC-CH Advanced Microscopy Facilities for Science Education Outreach, K. Jeffay, R. Superfine (Physics and Astronomy), G. Jones (Education), University of North Carolina at Chapel Hill

Instructional Technology Award, 1997, $29,750.

Support for the Real-Time Technology and Applications Symposium, K. Jeffay, Office of Naval Research, 1997, $7,000.

Research Experience For Undergraduates supplement to *Object Sharing Technology For Real-Time Systems*, J. Anderson, K. Jeffay, National Science Foundation, 1996, $10,000.

Infrastructure for Research in Collaborative Systems, S.F. Weiss, J.B. Smith, K. Jeffay, P. Dewan, P. D. Stotts, D.K. Smith, F.D. Smith, W. Oliver (U.S. Armed Forces Institute of Pathology), National Science Foundation, grant number CDA-9624662, August 1996, 5 years, $1,260,830.

Collaboration Bus: An Infrastructure for Supporting Interoperating Collaborative Systems, P. Dewan, K. Jeffay, H. Abdel-Wahab (Old Dominion University), P. D. Stotts, L. Nyland, J.B. Smith, J. Mchugh (Portland State University), J. Menges (Hewlett Packard), Advanced Research Projects Agency, grant number 96-06580 (High Performance Distributed Services Technology), 1996, $973,334.

Object Sharing Technology For Real-Time Systems, J. Anderson, K. Jeffay, National Science Foundation, grant number CCR 95-10156, 1996, $209,442.

Flexible Shared Windows, P. Dewan, K. Jeffay, National Science Foundation, grant number IRIS 95-08514, 1995, $343,811.

An ATM Testbed For Multimedia Networking and Computer-Supported Cooperative Work, K. Jeffay, IBM Corporation, 1995, $400,000.

An Examination of Flow and Congestion Control Mechanisms for Media Transmission in Collaborative Systems, K. Jeffay, IBM Corporation, 1995, $125,242.

Software Infrastructure for the Rapid Development of Interactive and Collaborative Educational Simulations, J.F. Prins, K. Jeffay, P.D. Stotts, L.S. Nyland, Advanced Research Projects Agency, grant number 95-36871 (Computer Aided Education and Training Initiative), 1995, $200,000.

A Proposal For Network Routers, K. Jeffay, IBM Corporation, Research Triangle Park, NC, 1994, $40,000.

System Support For Video Teleconferencing Across Local Area Networks, K. Jeffay, Intel Corporation, 1993, $253,385.

The Integration and Use of Digital Audio and Video in Desktop Computing Environments, K. Jeffay, IBM Corporation, 1993, $100,000.

Construction of a UNC-Tektronix MBONE Link, K. Jeffay, Tektronix Corporation, 1993, $10,000.

An Empirical Determination of the Limits of Human Perception of Properties of Digital Audio and Video Streams, K. Jeffay, University of North Carolina Research Council, 1993, $2,000.

Processor and Resource Allocation Problems in Hard-Real-Time Systems: Theory and Practice, K. Jeffay, National Science Foundation, RIA Award, 1991, $59,400.

Accommodating Continuous Media in a Local Area Network: An exercise in distributed real-time computing, K. Jeffay, University of North Carolina Junior Faculty Development Award, 1990, $3,000.

Building and Using a Collaboratory: A Foundation for Supporting and Studying Group Collaborations, J.B. Smith, F.D. Smith, P. Calingaert, K. Jeffay, J.R. Hayes, D. Holland, M. Lansman, National Science Foundation, grant number ICI-9015443, 1990, $946,000.

Enhanced program of research and teaching in communications software and distributed systems, K. Jeffay, IBM Corporation, 1989, $429,000.

The Design and Construction of Predictable Real-Time Systems, K. Jeffay, Digital Faculty Program Award, Digital Equipment Corporation, 1989, $180,000.

Invited Presentations

Keynote Addresses

Network Neutrality Considered Harmful,

PearlHacks, University of North Carolina at Chapel Hill, Chapel Hill, NC, March 2014.

Carolina Science Café, University of North Carolina at Chapel Hill, Chapel Hill, NC, May 2015.

Network Neutrality Considered Harmful, Revisiting the Quality-of-Service Morass

ACM International Workshop on Network and Operating System Support for Digital Audio and Video, Newport, RI, May 2006.

Rate-Based Resource Allocation Methods for Multimedia Computing

SPIE Multimedia Computing and Networking 2003, Santa Clara, CA, January 2003.

Network Support For Distributed Virtual Environments: The Tele-nanoManipulator

NCNI Advanced Networking Symposium, Research Triangle Park, NC, May 1999.

The Internet 2 Spring Members Meeting (presented jointly with R.M. Taylor), Washington, DC, April 1999.

Network Support For Distributed, Immersive, Virtual Environments

Workshop on Automated Control of Distributed Instrumentation, Beckman Institute for Advanced Science and Technology, University of Illinois at Urbana-Champaign, Urbana, IL, April 1999.

The Future of Networking: The Networking Revolution Has Yet to Begin

North Carolina Governor's Board of Science and Technology Retreat, Banner Elk, NC, September 1998.

Distinguished Lectures

The Synthetic Traffic Generation Problem: The least sexy problem in computer networking

University of Minnesota, Minneapolis, MN, January 2008.

The Effect of Active Queue Management on Web Performance: The Good, the Bad, and the Ugly

University of Nebraska, Lincoln, NE, November 2003.

University of Pennsylvania, April 2004.

The Evolution of Quality-of-Service on the Internet

Mälardalen University, Sweden, February 2001.

Colloquia

The Evolution of Quality-of-Service on the Internet

IEEE Computer Society Seattle Chapter, Seattle, WA, September 2005.

Modeling and Generating TCP Application Workloads

Georgia Institute of Technology, Atlanta, GA, September 2005.

Microsoft Research, Redmond, WA, September 2005.

Worcester Polytechnic Institute, Worcester, MA, July 2005.

Cisco Systems, RTP, NC, May 2005 (presently jointly with F. Hernandez-Campos and F.D.

Smith).

A Rate-Based Execution Abstraction For Embedded Real-Time Systems

University of Pennsylvania, April 2004.

How “Real” Can Synthetic Network Traffic Be?

University of Virginia, March 2004.

Non-Parametric Approach to Generation and Validation of Synthetic Network Traffic

Columbia University, New York, NY, January 2004.

Is Explicit Congestion Notification (ECN) Worthwhile?

Cisco Systems, San Jose, CA, October 2003.
Intel Architecture Labs, Hillsboro, OR, October 2003.

Tuning RED for Web Traffic: RED Considered Harmful?

Sprint Advanced Technology Research Laboratories, San Francisco, CA, March 2000.
University of Illinois at Urbana-Champaign, Urbana, IL, March 2000.

Internet Traffic: Measurement & Generation

Ganymede Software, Research Triangle Park, NC, June 1999. (Presented jointly with F.D. Smith.)

Lightweight Active Router-Queue Management for Multimedia Networking

University of Toronto, October 2000.
HP Laboratories, Palo Alto, CA, January 1999.
Sprint Advanced Technology Research Laboratories, San Francisco, CA, January 1999.

A Better-Than-Best-Effort Service For UDP: Lightweight Active Router-Queue Management for Multimedia Networking

University of Virginia, January 1999.

Congestion Control Mechanisms for Real-Time Communications on the Internet

IBM Networking Systems, Research Triangle Park, NC, November 1998.

A Better Than Best-Effort Forwarding Service For UDP

Microsoft Research, Redmond, WA, March 1998.
IBM Networking Systems, Research Triangle Park, NC, March 1998.

Multimedia Networking Research at UNC Chapel Hill

IBM Networking Systems, Research Triangle Park, NC, February 1998.
Lucent Technologies, Bell Laboratories, Holmdel, NJ, June 1997.

Design Principles for Distributed, Interactive, Virtual Laboratories

Mitsubishi Electric Research Laboratory, Cambridge, MA, January 1998.
Lucent Technologies, Bell Laboratories, Holmdel, NJ, June 1997.

The Performance of Two-Dimensional Media Scaling for Internet Videoconferencing

Mitsubishi Electric Research Laboratory, Cambridge, MA, January 1998.

Carnegie Mellon University, Pittsburgh, PA, March 1997.

Some Approaches to Enabling Real-Time Computation on Desktop Operating Systems

Honeywell Technology Center, Minneapolis, MN, October 1996.

Two-Dimensional Scaling Techniques for Adaptive, Rate-Based Transmission Control of Live Audio and Video Streams

Oregon State University, Corvallis, OR, May 1996.

A Router-Based Congestion Control Scheme For Real-Time Continuous Media

Intel Architecture Development Laboratory, Hillsboro, OR, May 1996.
NetEdge Inc., Research Triangle Park, NC, March 1996.
North Carolina State University, Raleigh, NC, March 1996.

A Hybrid Reservation-Based/Best-Effort Transmission Scheme For Real-Time Transmission of Multimedia Data

Intel Research Council, Hillsboro, OR, November 1995.

Operating System Support For Multimedia Computing

IBM Networking Systems, Research Triangle Park, NC, July 1995.

Network Support For Multimedia Computing

IBM Networking Systems, Research Triangle Park, NC, July 1995.

A Rate-Based Execution Abstraction For Multimedia Computing

Tektronix Computer Research Laboratory, Beaverton, OR, May 1996.
INRIA, Rocquencourt, France, December 1995.
AT&T Bell Laboratories, Murray Hill, NJ, July 1995.

A Theory of Rate-Based Scheduling

Carnegie Mellon University, Pittsburgh, PA, August 1994.

Demonstration of Videoconferencing Over Packet-Switched Internetworks

Global Lecture Hall of the U.S.-Russian Electronic Distance Education System/TELE-TEACHING '93, Trondheim, Norway (via satellite from Chapel Hill), August 1993.

Transport and Display Mechanisms For Multimedia Conferencing Across Packet-Switched Networks

York University, York, England, November 1993.
Purdue University, West Lafayette, IN, February 1993.
Intel Architecture Development Laboratory, Hillsboro, OR, February 1993.
Oregon Graduate Institute, Beaverton, OR, February 1993.
Second MCNC/NSF Packet Video Workshop, Research Triangle Park, NC, December 1992.
Carnegie Mellon University, Pittsburgh, PA, November 1992.

Multimedia and Conferencing Research at UNC-CH

IBM Multimedia Networking, Research Triangle Park, NC, September 1992.

System Support for Synchronous Collaboration

North Carolina Artificial Intelligence and Advanced Computing Symposium, Raleigh,

NC, March 1992.

Software Architectures for Predictable Real-Time Computer Systems

Carnegie Mellon University, Pittsburgh, PA, March 1992.

Operating System Requirements for Digital Audio and Video

MCNC/NSF Packet Video Videoconferencing Workshop, Research Triangle Park, NC, December 1991.

Some Experiments With Live Digital Audio and Video on a Local-Area Network

IBM European Networking Center, Heidelberg, Germany, November 1991.

Network and Operating System Support for Digital Audio and Video

Duke University, Durham, NC, November 1994.
Carnegie Mellon University, Pittsburgh, PA, April 1994.
Intel Corporation, Hillsboro, OR, December 1993.
IBM Networking Systems, Research Triangle Park, NC, June 1992.
IBM TJ Watson Research Center, Yorktown Heights, NY, March 1992.
University of Washington, Seattle, WA, February 1992.
Old Dominion University, Norfolk, VA, October 1991.

Systems Research in the UNC Collaboration Project

Intel Architecture Development Laboratory, Hillsboro, OR, May 1992.
University of Colorado, Boulder, CO, March 1991.

Some Deterministic Resource Allocation Problems in Real-Time Computer Systems

Research Triangle Institute, Research Triangle Park, NC, June 1990.
University of North Carolina at Chapel Hill, Department of Operations Research, Chapel Hill, NC, October 1989.

The Real-Time Producer/Consumer Paradigm: Towards Verifiable Real-Time Computations

Carnegie-Mellon University, Pittsburgh, March 1991.
Duke University, Durham, NC, June 1990.
Technical University of Denmark, Lyngby, Denmark, September 1989.
University of California at Davis, Davis, CA, April 1989.
University of Minnesota, Minneapolis, MN, April 1989.
Purdue University, West Lafayette, IN, April 1989.
University of North Carolina at Chapel Hill, Chapel Hill, NC, April 1989.
University of British Columbia, Vancouver, BC, Canada, April 1989.
Rice University, Houston, TX, March 1989.
University of Arizona, Tucson, AZ, March 1989.
Oregon Graduate Center, Beaverton, OR, March 1989.
Tektronix Computer Research Laboratory, Beaverton, OR, March 1989.
University of Colorado, Boulder, CO, March 1989.
IBM TJ Watson Research Center, Yorktown Heights, NY, March 1989.
IBM Systems Integration Division, Owego, NY, March 1989.
New York University, New York, NY, March 1989.
Bell Communications Research, Morristown, NJ, March 1989.

Concurrent Programming With Time

Olivetti Research Center, Menlo Park, CA, December 1987.
IBM TJ Watson Research Center, Yorktown Heights, NY, September 1987.
Columbia University, New York, NY, September 1987.

Short Courses

Trends in Congestion Control and Quality-of-Service: Active queue management on the Internet of the future

Mälardalen University, Sweden, February 2001.

Rate-Based Execution Models For Real-Time Multimedia Computing

Scuola Superiore Santa Anna, Pisa, Italy, September 1997.

Tutorials

Issues in Multimedia Delivery Over Today's Internet

IEEE International Conference on Multimedia Computing Systems, Austin, TX, June 1998.

Systems Issues in the Design and Realization of Desktop Videoconferencing Systems,

Second ACM International Conference on Multimedia, San Francisco, CA, October 1994.